EXTENDING 3D RECONSTRUCTION TO TEMPORAL AND MULTI-MODEL SENSOR DATA FOR PRECISION AGRICULTURE

A Dissertation
Presented to
The Academic Faculty

By

Jing Dong

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Interactive Computing

Georgia Institute of Technology

December 2018

EXTENDING 3D RECONSTRUCTION TO TEMPORAL AND MULTI-MODEL SENSOR DATA FOR PRECISION AGRICULTURE

Approved by:

Dr. Frank Dellaert, Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. Byron Boots, Co-Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. James Rehg
School of Interactive Computing
Georgia Institute of Technology

Dr. Patricio Vela
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Sudipta Sinha
Microsoft Research

Date Approved: August 23, 2018

ACKNOWLEDGEMENTS

I would like to thank everyone that has helped and supported me throughout the past five years in Tech. First of all, I would like to thank my advisor Prof. Frank Dellaert, for his guidance and support. I'm privileged to join Frank's group in 2013 and start a fascinating journey on robotics. Since then, Frank has helped me not only build up computer science skill set from scratch, but also provide impressive insight of research philosophy from high level. Beside research, I also benefit a lot from his valuable career development advice base on his experience in both academia and industry. Most importantly, his curiosity, optimism and his practice of life-ling learning has always been inspiring me to be a better researcher, and a better person. I would also like to thank my co-advior, Prof. Byron Boots. It is Byron that introduced me to machine learning research. While most of our work was not included in this thesis, we had a great time working together through those exciting projects, where Byron has dedicated many time in detailed guidance.

I'd like to also thank my committee members: Prof. James Rehg, Prof. Patricio Vela, and Dr. Sudipta Sinha. I'm honored to have them on my committee and very grateful for the time they have spent in providing valuable feedback and suggestions on my research.

My thanks also go to my collaborators at GTRI, Univeristy of Georgia, and Microsoft Research. I really want to thank Prof. Glen C. Rains and Prof. Gary McMurray for their guidance and support in the NRI project. Writing this thesis is impossible without their insight in agriculture and robotics, and their guidance and support in making the dataset. I'm also grateful for the valuable mentorship provided by Dr. Sudipta Sinha while I was interning at Microsoft Research, and the year after I came back to Georgia Tech. I really appreciate his mentorship and innovative insight to our research project. experimental and theoretical part of the project. I would like to thank Dr. Ranveer Chandra and James Pavis's support in Farmbeats project, and collecting the drone dataset.

I was privileged to collaborate with excellent researchers from Borg Lab and Georgia

Tech Robot Learning Lab in my Ph.D. period, in particular, Prof. Luca Carlone, Prof. Vadim Indelman, Zhaoyang Lv, Yong-Dian Jian, Duy-Nguyen Ta, Natesh Srinivasan, Siddharth Choudhary, Abhijit Kundu, Andrew Melim, Richard Roberts, Mustafa Mukadam, Xinyan Yan, and many other people. Thank you for being my community at GT!

In the end, I wish to thank my family who have always supported my dreams and my decision to do a Ph.D. In particular, I wish to thank my girlfriend Chuchu, for her inspiration, belief and love in me, which are a vital support for me to finish my Ph.D. I could not ask for a better partner in my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Crop monitoring is one of the most important tasks in precision agriculture, and to reduce cost, such task is often performed autonomously by unmanned aerial and ground vehicles. To capture 3D geometric information about crops, existing systems mostly use LIDAR, but LIDAR is expensive and there is a desire to replace it with cheaper sensors like monocular cameras coupled with techniques to obtain 3D reconstructions from 2D images. One of the major disadvantages of many existing 3D reconstruction algorithms is that they assume the scene is static, and they cannot be used to monitor crops growing over time. Moreover, many existing 3D reconstruction algorithms are not designed to handle multi-spectral or hyper-spectral images, which are commonly used in precision agriculture to recover information that cannot be seen by naked eye.

In this work I propose a full pipeline for building 3D reconstructions from temporal and multi-modal image sequences to use in precision agriculture applications. The three major technical contributions are: (1) 3D reconstruction for low-cost systems enabled by Gaussian process based continuous-time SLAM, (2) spatio-temporal 4D reconstruction to enable the monitoring of crops over time, and (3) weakly-supervised learning of local image descriptors between multiple image modalities. I also collected a multi-year growing crop dataset in order to evaluate the performance of the proposed pipeline.

CHAPTER 1
INTRODUCTION

1.1 Motivation

The growth and final yield of crops are threatened by many biotic and abiotic factors, such as pests, weeds, lack/excess water and nutrition. Although pesticide, fertilizer, and irrigation system are widely used in modern agriculture, research shows that farmers are still suffering about 20%-40% losses of final yield worldwide [1]. Why farmers still suffer yield losses with help from pesticide/fertilizer/irrigation? The main reason is not the tools, but decision making. A farm does not share the same condition everywhere in the field in general. Soil, nutrition condition and micro-climate can differ over the field. If a farmer makes one decision and apply the same action over the whole field, the growth may not be optimal everywhere, thus yield loss happens.

The major methodology to overcome the issue is *precision agriculture* [2]. The generic definition of precision agriculture is "that kind of agriculture that increases the number of (correct) decisions per unit area of land per unit time with associated net benefits" [2]. With increased spatial resolution of decision making process, most areas of the field will reach the optimal condition for crop growing, and the final yield will be maximized.

To increase number of decisions over the field, more information of the field with increased spatial resolution is needed from various types of sources. Field trials have shown that using soil sensor measurements to vary amount of water to use in the field increases the final yield by 45% meanwhile reducing water usage by 35% [3]. But due to high cost of deploying soil sensors at high spatial resolution and manual labor to process sensor reading, only less than 20 percent farmers in US can afford such sensors, as reported by USDA [4]. Aerial and satellite imageries are good sources of field and crop information, but they all

have limited both spatial and temporal resolutions. Color and spectral satellite images have been collected and used for yield prediction purpose [5, 6], and USDA provides high resolution images captured by manned aircrafts. But due to the limited flexibility of satellite and manned aircraft flights, aerial and satellite imageries are available at relatively low and uncontrolled frequencies, like few updates per year, which may not enough for farmers to make timely decisions during crop growing seasons.

Recently, crop monitoring with unmanned aerial vehicles (UAVs) [7, 8, 9, 10, 11] and unmanned groundvehicles (UGVs) [12, 13, 14, 15, 16] have drawn increasing attentions from both precision agricultural and robotics communities, since its flexibility to provide sensor data at high spatial and temporal resolutions, and its low cost achieved by saving human labor. In [9] a system is built with a LIDAR scanner, a stereo camera, an IMU plus a GPS, and the system is mounted on a multi-rotor UAV to obtain 3D occupancy grids of trees. To reduce system cost, a 2D laser scanner other than 3D LIDAR is mounted on a multi-rotor UAV to obtain tree heights in [8]. To further reduce system cost, instead of using 2D laser scanner or 3D LIDAR, researchers [12, 10, 11] use lower cost camera as the main sensor on the UAV, to obtain crop information solely from analyzing 2D images [12, 11] or by running 3D reconstruction on images and get 3D crop information [10].

3D reconstruction is a very useful tool in precision agriculture to get 3D geometric information of crops from 2D images. Although computer vision techniques on 2D images have been used in crop monitoring and yield estimation for years [12, 15, 17, 18], 3D models provide more geometric information and they are recently preferred in precision agriculture, including forest monitoring [13], tree specie identification [19], tree counting in orchards [14, 16], and crop height measurement [20, 21]. But many of the existing systems rely on LIDARs to get 3D crop information [13, 19, 14, 16, 20, 21] , which are too expensive to be widely used by every farmers. There are also existing papers using Kinect depth sensor [22] and RGB color camera [23, 24] as input, but they all need artificial illumination to work properly, so they need special mobile shields to cover the cameras, or

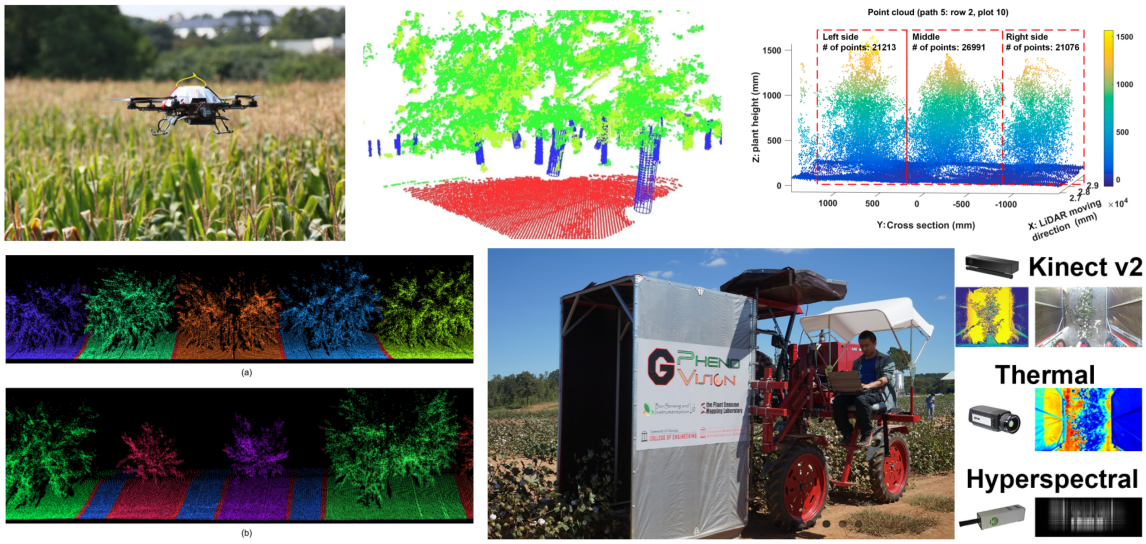


Figure 1.1: Representative works of using 3D information in crop mornitoring (from top to bottom, left to right): using UAV equipped with a 2D laser scanner to measure crop height [8]; using 3D LIDAR to estimate volume of wood in forest [13]; crop height estimation from 3D point cloud from LIDAR [21]; LIDAR-Based tree recognition in orchards [16]; capturing cotton depth and multi-spectual images by mobile shield to avoid sunlight interference [22].

the systems have to operate during night. To make the system affordable and to simple-to-use to farmers, we want to obtain 3D reconstruction of field and crops from regular color images taken by UAVs or UGVs under natural illumination.

Although the major technique to recover 3D structure from a set of 2D images is structure from motion (SfM) [25, 26], which is a well-studied topic, but most SfM algorithms assume that the scene is *static*, which is not the case in crop monitoring, since we need to reconstruct and model the *growth* of the crops. Thus the crop monitoring task leads to a research problem of 3D reconstruction of *dynamic* scenes, which leads to the problem *4D reconstruction* (3D + time). Although there are several existing works about the topic 4D reconstruction [27, 28, 29, 30, 31], they have various constraints that prevent the methods to be directly applied to crop monitoring applications, which leads to open research questions.

Images within different wavelength ranges provide lots of information that cannot be seen by the naked eye, and these information are particularly important in agriculture,

since different types of pigment have different absorption distributions in electromagnetic spectrum, which can help us identity the health of crops. For example, chlorophyll has strong absorption in the red and blue portions of the electromagnetic spectrum [32]. Hyper-spectral and multi-spectral cameras are used to capture the imageries with different wavelength information. Low cost hyper-spectral and multi-spectral cameras have been commercialized for years, and Hyper-spectral and multi-spectral imageries are taken from various platforms, including satellite, drones and group vehicles. These platforms have been used in various crop monitoring tasks, including recent studies on cotton [33, 22] and blueberry [34]. Normalized difference vegetation index (NDVI) is one the most important application of multi-spectral image information in crop monitoring, since it is an indicator for vegetation density and health condition, and it can be also used to estimate other vegetation health index like leaf area index [35]. NDVI across different seasons [36] and different years [37, 38] has been studies on satellite imageries.

However, most existing SfM implementations are not compatible with mixed input from different image modalities, which is a common case in precision agriculture. To enable SfM techniques with combining different modalities from multi-spectral images, *data association* between different image modalities is needed, e.g. registering images between modalities. Registering images between modalities is neither a new need (The NDVI need be calculated by the visible red and near infrared light as the same point), nor new research topics [39, 40, 41, 42]. However, due to the target image types of most existing methods (satellite or medical imageries), these techniques do not directly apply to 3D reconstruction applications. Therefor, there are demands of combining 3D reconstruction techniques with imageries of multiple modalities, like spectral imageries.

In this thesis I propose a full multi-sensor 3D reconstruction pipeline, which take temporal image sequences from multi-model image sensors, plus various other sensor data such as inertial measurement unit (IMU) and GPS data, output a series of registered 3D models with temporal and multi-model sensor information. A temporal and multi-model

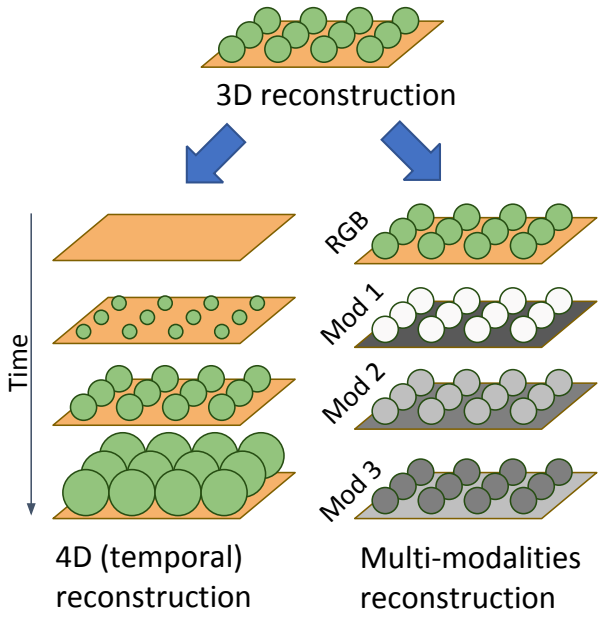


Figure 1.2: Concept of temporal (left) and multi-model (right) 3D reconstruction.

3D reconstruction consists multiple 3D reconstructions, each one reconstructs the field a single time and uses images from one type of image sensors. All 3D reconstructions are well registered into a single coordinate frame, so we can easily recover 3D geometric information of crops of one place at different time and wavelength. The concept of the temporal and multi-model 3D reconstruction is shown in Figure 1.2.

1.2 Thesis statement and claims

The outcomes of this thesis support the following thesis statement:

Extending 3D reconstruction to spatio-temporal reconstruction and multi-model 3D reconstruction gives us the ability to capture time-series and multi-spectral 3D information of crops, enabling better crop monitoring in precision agriculture.

In particular, I unpack this thesis into three separate sub claims:

1. Monocular 3D reconstruction method obtains highly accurate and large scale 3D reconstructions from various low-cost sensor data, and the 3D reconstructions are useful to estimate 3D information of crops.

2. Spatio-temporal (4D) reconstruction method obtains highly accurate and large scale spatio-temporal reconstructions from various low-cost sensor data, which are collected at time sequences, and the spatio-temporal reconstructions are useful to estimate 3D information of crops in temporal sequences.

3. Weakly-supervised local image descriptor and alignment learning obtains robust image registration of different modalities from input without annotation, enables getting multi-spectral 3D information of crop with little human labor.

1.3 Contributions

There are four major contributions of this thesis:

1. A *large scale*, *low cost*, and *flexible* monocular 3D reconstruction pipeline using low-cost sensor input, including a general Gaussian process (GP) based continuous-time trajectory optimization for fusion of asynchronous multiple sensors.

2. An *accurate*, *low cost*, and *flexible* spatio-temporal (4D) reconstruction pipeline, which models continuously changing scenes in 3D.

3. A *weakly-supervised*, *general*, and *robust* local image descriptor and alignment learning for cross-modality image registration.

4. A multi-year large scale field dataset collected by aerial and ground vehicles for evaluating crop monitoring applications.

The thesis is organized in the following way: Chapter 3 describes the proposed monocular 3D reconstruction pipeline including the proposed GP based continuous-time trajectory estimation method; Chapter 4 describes the proposed spatio-temporal reconstruction pipeline; Chapter 5 describes the proposed weakly-supervised cross-modality image registration learning. Also, two datasets are collected for evaluation purpose, one is a multi-year field dataset collected by ground vehicles, the other one is a multi-spectral dataset collected

by unman aerial vehicles. Both datasets are first introduced in Chapter 2 since they are used in following chapters.

CHAPTER 2

DATASETS

Although there are existing datasets collected with both large scale spatial and temporal information [43, 44, 45], however, all of these datasets were collected in urban environments (mostly on the streets), targeting autonomous driving applications, and are not suitable for evaluating algorithms for precision agriculture applications.

While working on this thesis, I collected two field datasets for the purpose of evaluation. One dataset is collected by ground vehicles at Tifton, Georgia with GTRI and the University and Gerogia, the other one was collected by aerial vehicles near Seattle Washington with Microsoft Research. The ground vehicle dataset was collected with multiple sensors (RGB camera, GPS, IMU, etc.), and used to evaluate spatio-temporal reconstruction. The drone dataset is collected with a multi-spectral camera, and used to evaluate multi-modal image registration.

2.1 Ground dataset collected at Tifton, GA

2.1.1 Equipment

I worked with GTRI and the University and Gerogia, to set up a ground vehicle equipped with a sensor box to collect a 3 years (from 2016 to 2018) field dataset at Tifton, GA [46]. In particular, I built sensor box that contains (1) a Point Grey monocular global shutter camera, streams 1280×960 color images at 7.5Hz, (2) a 9-DOF IMU with magnetometer, acceleration and angular rate are streamed at 167Hz, and magnetic field data is streamed at 110Hz, (3) a high accuracy RTK-GPS, and a low accuracy GPS, both of them stream latitude, longitude and absolute height data at 5Hz, and (4) a computer with data storage and remote control access. The sensor combo is mounted on a ground vehicle which is

shown in Figure 2.1.

2.1.2 Field datasets

On the course of 2016, I recorded a complete season of peanut growth, which started May 25, and completed Aug 22, right before harvest. The size of the field is about 150m$\times$120m, and the map of the field is shown in Figure 2.2. The data collection had a total of 23 sessions over 89 days, approximately two per week, with a few exceptions due to severe weather conditions. Example images of different dates are shown in Fig. 2.3. Each session lasted about 40 minutes, and consisted of the tractor driving about 3.8 km in the field. At the end of the peanut season, total of 140 GB data were collected, include 15 hours tractor runs, 70 km total drives, more than 350 thousands images, and 450 thousands GPS coordinates are recorded. In addition to sensor data, ground truth crop properties (height and leaf chlorophyll) at 47 sampling sites in the field were measured weekly by human operators.

The 2017 dataset was collected at the same field with year 2016, but with different ground vehicle paths Total of 23 sessions over 98 days were collected, but due to corrupted RTK-GPS data in some of the sessions, only 17 sessions are good for later data analysis. Each session lasted about 50 minutes, and consisted of the tractor driving about 5.2 km in the field. Example images of different dates are shown in Fig. 2.4. At the end of the peanut season, total of 110 GB good data were collected, include 14 hours tractor runs, 88 km total drives, more than 370 thousands images, and 500 thousands GPS coordinates are recorded.

The 2018 dataset was collected at a different field with 2016 and 2017, which is smaller than previous years (about 1/3 size of the 2016/2017 field). Total of 4 sessions over 34 days were collected before this thesis is finished. Each session lasted about 20 minutes, and consisted of the tractor driving about 1.6km in the field. Example images of different dates are shown in Fig. 2.5.

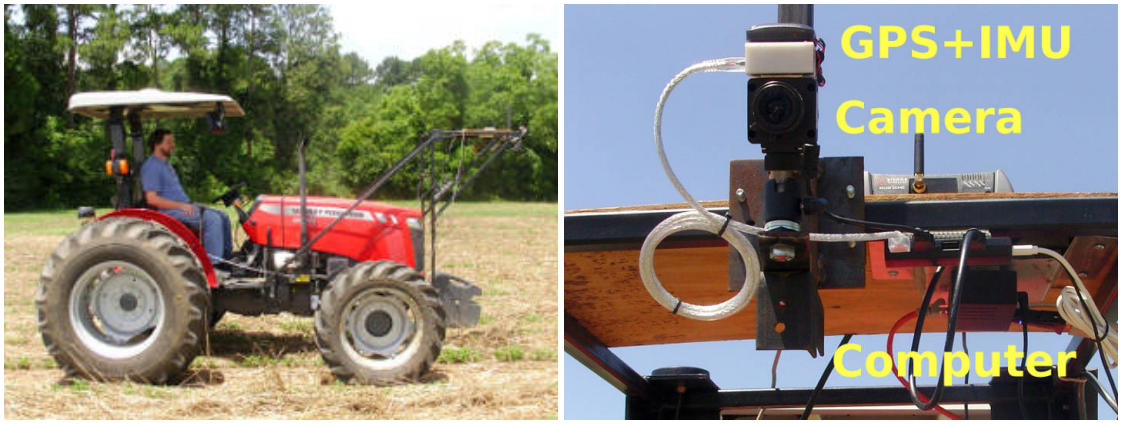
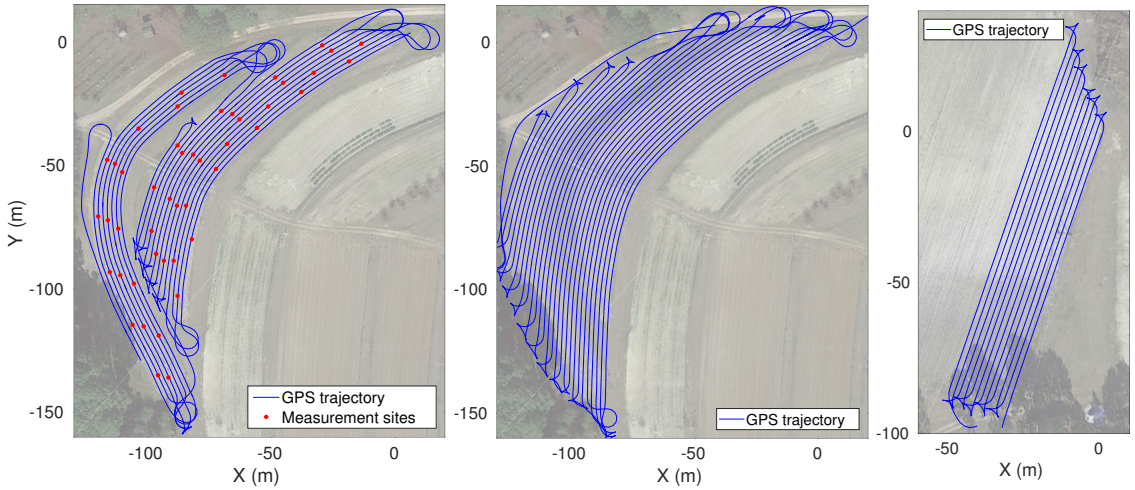


Figure 2.1: Left: the tractor used to mount sensor box and collect dataset. Right: the sensor box, include a monocular color camera, an PX4 IMU module with a low precision GPS, a RTK-GPS module (not shown in image), an Odroid U3 computer, a 256GB SSD hard drive, and a cell data module for remote control.



(a) Year 2016 (b) Year 2017 (c) Year 2018

Figure 2.2: The ground vehicle RTK-GPS trajectories of field datasets collected from 2016 to 2018, overlaid on Google Map, shown from left to right. For 2016 dataset, the manual height and leaf chlorophyll sampling locations are marked as red dots.

Figure 2.3: Eight sample image frames taken at approximately the same location in the field in 2016, dates taken are marked on images.

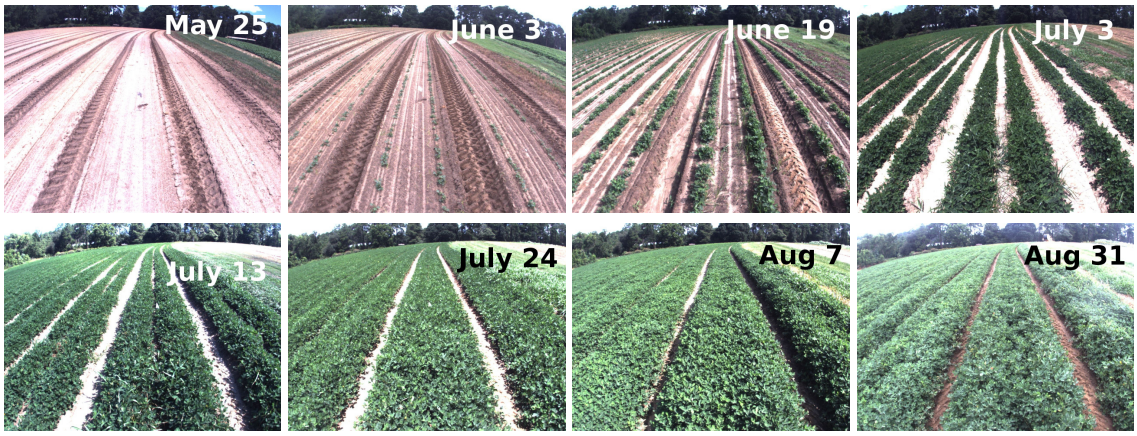


Figure 2.4: Eight sample image frames taken at approximately the same location in the field in 2017, dates taken are marked on images.

Figure 2.5: Four sample image frames taken at approximately the same location in the field in 2018, before the thesis is done, dates taken are marked on images.

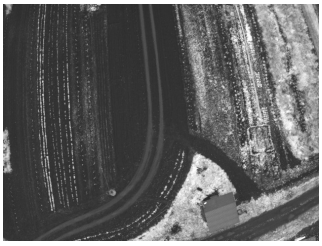
2.2 Drone dataset collected near Seattle, WA

The drone dataset was collected in a field near Seattle, WA in collaboration with Microsoft Research. The vehicle is a 3DRobotics quadrotor, shown in Fig. 2.6, equipped a MicaSense Parrot Sequoia multi-spectral camera. The Sequoia multi-spectral camera have 5 independent lens and CMOS sensors, as shown in Fig. 2.6: One is regular RGB camera, four are narrow band cameras. The four narrow band cameras are sensitive to near-infrared (800nm), red-edge (720nm), red (650nm) and green (550nm) light, and wavelength bands of each channel are all smaller than 30nm. Five cameras have similar field of view, and their optical axises are roughly aligned to the same direction. But the camera is neither calibrated, nor the optical axises are aligned exactly due to manufacturing errors, so the output images are not aligned or registered. Fig. 2.7 shows an example image set.

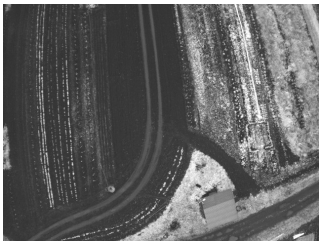
During data collection, the drone was set up to fly 30 meters above the field, and follow a pre-programmed path. Due to data license, the dataset available for evaluation in this thesis is limited 50 non-consecutive image sets selected from multiple data collection sessions. This dataset will be used in evaluation section of Chapter 5.

Figure 2.6: Left: the drones used to collected dataset (photo courtesy of James Pavis). Right: the Sequoia multi-spectral camera used to capture multi-spectral images, text indicates band of each lens.

(a) RGB

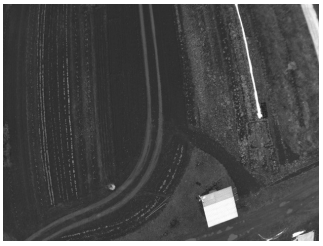


(b) Near-infrared (800nm)



(c) Red edge (720nm)

(d) Red (650nm)



(e) Green (550nm)

Figure 2.7: An example image set of RGB and four spectrum, captured by Sequoia camera at 30m height.

CHAPTER 3

3D RECONSTRUCTION OF DATA FROM LOW-COST SENSORS

3.1 Introduction and claims

As discussed in Chapter 1, in most current precision agriculture systems, 3D information is captured by LIDARs which are very expensive. Kinect depth sensors have also been used in precision agriculture [22], but due to structured lighting pattern getting lost in the ambient infrared light, Kinect sensors have to be used during night time or in a covered shield in outdoor applications.

Structure from motion (SfM) and its cousin, simultaneous localization and mapping (SLAM), have been extensively used in 3D reconstruction from images. SfM is mostly used for offline reconstruction in which images are captured with no particular order, even not by the same camera (e.g. Google image searching result); SLAM is mostly used for online applications, in which image sequences or videos are from a single (or a few) camera with known time order. Although the 3D reconstruction has not to be done online by proposed method, our dataset has online nature: all images come from the same calibrated camera and they are all in a sequence with known order and time interval. So we propose a multi-sensor SLAM pipeline for 3D reconstruction in the field, and use it to process datasets offline (not on the vehicle).

One of the major difficulties in designing a multi-sensor SLAM pipeline is how to fuse information from multiple sensors, especially when messages from different sensors are not hardware-synchronized, i.e., when messages arrive from different ports asynchronously. For the SLAM pipeline which targets to 3D reconstruction purpose, we want to estimate transformations of the camera at the time steps when images are taken, using sensor input not limited to images, e.g. GPS coordinates and IMU readings. But what if the other sensor

are not read exactly at the time when images are taken? Hardware synchronization can be applied to avoid such issue by trigging all sensor reading by a single signal, but all sensors must support external trigging and synchronization hardware have to be installed, which may not be the case in low-cost systems, such as our data collection system in Chapter 2.

Continuous-time trajectory representations provide solutions for the difficulty mentioned above. Unlike discretized trajectory representations that are commonly used in SLAM, continuous-time representations can be used to query robot states over the trajectories at any time of interest, and perform sensor fusions with asynchronous sensor data.

In this chapter I propose a 3D reconstruction pipeline to obtain 3D models of fields using a low-cost monocular camera as the main sensor to get 3D structure, although a stereo camera pair is also compatible to this framework. In this section the proposed 3D reconstruction pipeline has two parts: first I propose a general Gaussian process (GP) based continues-time SLAM framework for fusion of data from multiple asynchronous sensors; second I propose a monocular SLAM pipeline for low-cost precision agriculture systems.

My claims for the proposed 3D reconstruction pipeline are:

- *Low cost*: the proposed 3D reconstruction pipeline works for low-cost precision agriculture systems, by 3D reconstruction using a monocular camera and avoiding expensive LIDAR.

- *Flexible*: the proposed 3D reconstruction pipeline is plug-and-play to any type of sensor, and no hardware synchronization is needed, by using the proposed general GP-based continues-time SLAM framework.

- *Large scale*: the proposed 3D reconstruction pipeline works for large fields in real farms, with limited memory usage, because of system level contribution combining sensor fusion by factor graph and structure-less vision factor.

This chapter summarizes and extends materials from my previous publications [47, 48, 49].

3.2 Related work

SfM and SLAM SfM has been used for offline 3D reconstruction for many years. Early SfM methods are matrix factorization based method [50], but people have moved to optimization based method called bundle adjustment [51] for years, since compare to matrix factorization based methods, bundle adjustment works more efficiently for large scale problems, like problems of city scale [25, 26].

Another widely used method for 3D reconstruction is monocular SLAM, which is mostly used in online scenarios. Most early works of monocular SLAM are extended Kalman filter (EFK) based [52], then people move to optimization and factor graph based methods [53]. Some recent works using direct method e.g. LSD-SLAM [54] and DSO [55] or semi-direct method like SVO [56]. In the multi sensor 3D reconstruction pipeline, factor graph based method [53] is chosen, since factor graph has been widely used in sensor fusion [57], and can be easily extended to multi-sensor cases.

Continuous-time SLAM Continuous-time trajectory representations address two problems in SLAM: (1) when sensors continuously measure environment and output data, there is no fixed time stamp for sensor output, thus this data can be hardly supported by SLAM methods using discrete-time trajectories. Example sensors include spinning LIDAR and rolling-shutter cameras. (2) When sensor measurements arrive asynchronously, for example in the agriculture dataset, camera images come at 7.5Hz and GPS measurements come at 5Hz. It's hard to estimate camera poses with help from GPS with discrete-time trajectory, since data from both sensors are not arrived synchronously. Several popular continuous-time trajectory representations include linear interpolation [58, 59, 60], splines [61, 62, 63, 64, 65, 66], hierarchical wavelets [67] and Gaussian processes [68, 69, 70, 71]. In this section GP-based continues-time SLAM methods [68, 69, 70, 71] are used and extent, and a more detailed literature review about GP-based continuous-time SLAM methods will be given in next section.

GP-based continues-time SLAM A Gaussian process (GP) is a probabilistic non-parametric representation of continuous-time trajectories, which has been used in state estimation and SLAM in recent years. Tong *et al.* [68] shows the trajectory estimation problems can be translated to GP regressions, called simultaneous trajectory estimation and mapping (STEAM), which is a continuous-time extension of SLAM. This approach is soon extended to full 3D problems [72]. By allowing various GP priors on robot trajectories, this approach is a general approach to solve various types of trajectory estimation, but it's also quite expensive in computation time due to the dense inverse kernel matrices.

The sparsity properties of the linear systems underlying SLAM problems has been well studied [51, 53, 73], and it is the key to keep scalability for many optimization-based SLAM algorithms. By applying appropriate GP prior on trajectories, Barfoot *et al.* [69] first found the inverse kernel matrices cab be exactly sparse, leading to efficient solution of GP regression. This approach is further extended to GP priors driven by nonlinear time-varying stochastic differential equations (NTV-SDEs) [70], and incremental GP regression frameworks [74]. But the major drawback of these approaches is they all require the system state is in vector space, which is not always viable in general trajectory estimation problems. For example, there is no canonical coordinate exist for rigid-body rotation in 3D space without singularity (Euler angles) and extra constrain (quaternions).

In this work the sparse GP approach [69] is extended to general matrix Lie groups [75], extending the approach to a much more general setting. By extending such approach to matrix Lie groups, the trajectory estimation problems could be solved by continuous-time GP approaches will be more general. For example, an attitude and heading reference system (AHRS) [76] can be treated as trajectory estimation on the special orthogonal group SO(3), and 3D trajectory estimation for a monocular camera without scale information can be treated as trajectory estimation on the similarity transformation group Sim(3) [54]. Sparse GP regression for STEAM [69] has been extended to the special Euclidean group SE(3) in Anderson et al. [71], and this work is a further extension.

3.3 Sparse Gaussian processes on Lie groups for continues-time SLAM

3.3.1 Preliminaries

Problem Definition

In this section I consider the problem of continuous-time trajectory estimation, in which a continuous-time system state $\boldsymbol{x}(t)$ is estimated from observations [68]. The system model is described as

$$\boldsymbol{x}(t) \sim \mathcal{GP}(\boldsymbol{\mu}(t), \boldsymbol{\mathcal{K}}(t, t')) \tag{3.1}$$

$$\mathbf{z}_i = \mathbf{h}_i(\boldsymbol{x}(t_i)) + \mathbf{n}_i, \mathbf{n}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_i), \tag{3.2}$$

where $\boldsymbol{x}(t)$ is represented by a Gaussian Process (GP) with mean $\boldsymbol{\mu}(t)$ and covariance $\boldsymbol{\mathcal{K}}(t, t')$. A measurement $\mathbf{z}_i$ at each time t_i is obtained by the (generally nonlinear) discrete-time measurement function $\mathbf{h}_i$ in Eq. (3.2) and assumed to be corrupted by zero-mean Gaussian noise with covariance $\boldsymbol{\Sigma}_i$.

Maximum a Posteriori Estimation

The maximum a posteriori (MAP) estimate of the trajectory can be computed through method Gaussian Process Gauss-Newton (GPGN) [68]. We first write down the objective function, with assumption that there are M observations and the definitions of following

terms

$$\boldsymbol{x} \doteq \begin{bmatrix} \boldsymbol{x}(t_1) \\ \vdots \\ \boldsymbol{x}(t_M) \end{bmatrix}, \boldsymbol{\mu} \doteq \begin{bmatrix} \boldsymbol{\mu}(t_1) \\ \vdots \\ \boldsymbol{\mu}(t_M) \end{bmatrix}, \boldsymbol{\mathcal{K}} \doteq [\boldsymbol{\mathcal{K}}(t_i, t_j)]\Big|_{ij, 1 \leq i, j \leq M},$$

$$\mathbf{z} \doteq \begin{bmatrix} \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_M \end{bmatrix}, \mathbf{h}(\boldsymbol{x}) \doteq \begin{bmatrix} \mathbf{h}_1(\boldsymbol{x}_1) \\ \vdots \\ \mathbf{h}_M(\boldsymbol{x}_M) \end{bmatrix}, \boldsymbol{\Sigma} \doteq \begin{bmatrix} \boldsymbol{\Sigma}_1 & & \\ & \ddots & \\ & & \boldsymbol{\Sigma}_M \end{bmatrix},$$

the MAP estimation can be written as

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmax}} \left\{ \frac{1}{2} \parallel \boldsymbol{x} - \boldsymbol{\mu} \parallel^2_{\boldsymbol{\mathcal{K}}} + \frac{1}{2} \parallel \mathbf{h}(\boldsymbol{x}) - \mathbf{z} \parallel^2_{\boldsymbol{\Sigma}} \right\}, \tag{3.3}$$

where $\|\|_{\boldsymbol{\Sigma}}$ is Mahalanobis distance defined as $\parallel \mathbf{x} \parallel^2_{\boldsymbol{\Sigma}} \doteq \mathbf{x}^\top \boldsymbol{\Sigma}^{-1} \mathbf{x}$. MAP estimation is therefore translated into a nonlinear least square optimization problem.

A Gauss-Newton non-linear optimization approach is used to solve this nonlinear least squares problem. By linearizing the measurement function $\mathbf{h}_i$ around a linearization point $\overline{\boldsymbol{x}}_i$, we obtain

$$\mathbf{h}_i(\overline{\boldsymbol{x}}_i + \delta \boldsymbol{x}_i) \approx \mathbf{h}_i(\overline{\boldsymbol{x}}_i) + \mathbf{H}_i \delta \boldsymbol{x}_i, \mathbf{H}_i \doteq \frac{\partial \mathbf{h}_i}{\partial \boldsymbol{x}}\Big|_{\overline{\boldsymbol{x}}_i}, \tag{3.4}$$

in which $\mathbf{H}_i$ is the Jacobian matrix of measurement function (3.2) at linearization point $\overline{\boldsymbol{x}}_i$. By defining $\mathbf{H} \doteq \operatorname{diag}(\mathbf{H}_1, \ldots, \mathbf{H}_M)$, A linearized least squares problem is got around linearization point $\overline{\boldsymbol{x}}$

$$\delta \boldsymbol{x}^* = \underset{\delta \boldsymbol{x}}{\operatorname{argmax}} \left\{ \frac{1}{2} \parallel \overline{\boldsymbol{x}} + \delta \boldsymbol{x} - \boldsymbol{\mu} \parallel^2_{\boldsymbol{\mathcal{K}}} + \frac{1}{2} \parallel \mathbf{h}(\overline{\boldsymbol{x}}) + \mathbf{H} \delta \boldsymbol{x} - \mathbf{z} \parallel^2_{\boldsymbol{\Sigma}} \right\}. \tag{3.5}$$

The GPGN algorithm starts from some initial guess of $\overline{\boldsymbol{x}}$, then, at each iteration, the optimal perturbation $\delta \boldsymbol{x}^*$ is found by solving linear system

$$(\boldsymbol{\mathcal{K}}^{-1} + \mathbf{H}^\top \boldsymbol{\Sigma}^{-1} \mathbf{H}) \delta \boldsymbol{x}^* = \boldsymbol{\mathcal{K}}^{-1} (\boldsymbol{\mu} - \overline{\boldsymbol{x}}) + \mathbf{H}^\top \boldsymbol{\Sigma}^{-1} (\mathbf{z} - \mathbf{h}). \tag{3.6}$$

and updating the solution by $\overline{\boldsymbol{x}} \leftarrow \overline{\boldsymbol{x}} + \delta\boldsymbol{x}^*$ until convergence.

The information matrix $\boldsymbol{\mathcal{K}}^{-1}$ in Eq. (3.6) encodes the GP prior information, and $\mathbf{H}^\top\boldsymbol{\Sigma}^{-1}\mathbf{H}$ represents information from the measurements. $\mathbf{H}^\top\boldsymbol{\Sigma}^{-1}\mathbf{H}$ is block-wise sparse in most SLAM problems [53], but $\boldsymbol{\mathcal{K}}^{-1}$ is not usually sparse for most commonly used kernels. In the following a GP prior with sparse inverse kernel structure will be defined, and the sparse structure will be exploited to efficiently solve the linear system.

3.3.2 Sparse GP Priors for Trajectory Estimation

A class of exactly sparse GP priors for trajectory estimation were proposed by Barfoot et al. [69]. Unfortunately, only vector-valued system states are correctly handled by this approach. In this section the approach in [69] for vector spaces will be briefly revisited first, then the approach will be extent to two types of Lie groups, the special orthogonal group SO(3) and the special Euclidean group SE(3).

GP Priors for Vector Space

Here we consider GP priors for vector-valued system states $\boldsymbol{x}(t)$ generated by linear time-varying stochastic differential equations (LTV-SDEs) [69]

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}(t)\boldsymbol{x}(t) + \mathbf{u}(t) + \mathbf{F}(t)\mathbf{w}(t), \tag{3.7}$$

where $\mathbf{u}(t)$ is the known system control input, $\mathbf{w}(t)$ is white process noise, and both $\mathbf{A}(t)$ and $\mathbf{F}(t)$ are time-varying system matrices. The white process noise is represented by

$$\mathbf{w}(t) \sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_C\delta(t - t')), \tag{3.8}$$

where $\mathbf{Q}_C$ is the power-spectral density matrix, which is a hyperparameter [70], and $\delta(t-t')$ is the Dirac delta function. The mean and covariance of LTV-SDE generated GP are

$$\boldsymbol{\mu}(t) = \boldsymbol{\Phi}(t, t_0)\boldsymbol{\mu}_0 + \int_{t_0}^{t} \boldsymbol{\Phi}(t, s)\mathbf{u}(s)ds \tag{3.9}$$

$$\begin{aligned}\boldsymbol{\mathcal{K}}(t, t') &= \boldsymbol{\Phi}(t, t_0)\boldsymbol{\mathcal{K}}_0\boldsymbol{\Phi}(t', t_0)^\top \\ &+ \int_{t_0}^{\min(t,t')} \boldsymbol{\Phi}(t, s)\mathbf{F}(s)\mathbf{Q}_C\mathbf{F}(s)^\top\boldsymbol{\Phi}(t', s)^\top ds\end{aligned} \tag{3.10}$$

where $\boldsymbol{\mu}_0$ is the initial mean value of first state, $\boldsymbol{\mathcal{K}}_0$ is the covariance of first state, and $\boldsymbol{\Phi}(t, s)$ is transition matrix.

In [69] it is proved that if the system is generated by the LTV-SDE in Eq. (3.7), the inverse covariance matrix $\boldsymbol{\mathcal{K}}^{-1}$ is block-tridiagonal.

The *constant-velocity* GP prior is generated by a LTV-SDE with white noise on the acceleration and has previously been used in trajectory estimation [68, 69, 71].

$$\ddot{\mathbf{p}}(t) = \mathbf{w}(t), \tag{3.11}$$

where $\mathbf{p}(t)$ is the N-dimensional vector-valued position (or pose) variable of trajectory, if the system has N degrees of freedom. To convert this prior into the LTV-SDE form of Eq. (3.7), a Markov system state variable is declared

$$\boldsymbol{x}(t) \doteq \begin{bmatrix}\mathbf{p}(t) \\ \dot{\mathbf{p}}(t)\end{bmatrix}, \tag{3.12}$$

The prior in Eq. (3.11) then can easily be converted into a LTV-SDE in Eq. (3.7) by defining

$$\mathbf{A}(t) = \begin{bmatrix}\mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0}\end{bmatrix}, \quad \mathbf{u}(t) = \mathbf{0}, \quad \mathbf{F}(t) = \begin{bmatrix}\mathbf{0} \\ \mathbf{1}\end{bmatrix}. \tag{3.13}$$

GP Priors on SO(3)

Before discussing sparse GP priors for general Lie groups, two specific examples, SO(3) and SE(3), will be discussed. The purpose to discuss these two specific groups is they have more intuitive Lie algebra structure (which will be formally defined later), and make the extension easier to understand.

The *Special Orthogonal Group* SO(3), is a matrix Lie group and represents 3D rotation matrices $\mathbf{R}$ defined by $\{\mathbf{R} \in \mathbb{R}^{3\times 3} : \mathbf{R}\mathbf{R}^\top = \mathbf{I}, \det(\mathbf{R}) = 1\}$. The continuous-time trajectory is then represented by the function $\mathbf{R}(t)$ that maps time to rotation matrices.

The relation between rotation and *body-frame angular velocity* is given by [77, p.52]

$$\dot{\mathbf{R}}(t) = \mathbf{R}(t)_b\boldsymbol{\omega}(t)^\wedge, \tag{3.14}$$

where ${}_b\boldsymbol{\omega}(t)$ is the body-frame angular velocity (the subscript b means the angular velocity is defined in body-frame), and $\wedge$ operator constructs a 3×3 skew symmetric matrix from a vector in $\mathbb{R}^3$

$$\boldsymbol{\omega}^\wedge = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}^\wedge = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{3.15}$$

Assume that body-frame angular velocity is constant (*constant-velocity prior*), and the body-frame angular acceleration is corrupted by white noise

$${}_b\dot{\boldsymbol{\omega}}(t) = \mathbf{w}(t), \tag{3.16}$$

then the nonlinear SDE that represents this prior is

$$\dot{\boldsymbol{x}}(t) = \frac{d}{dt}\begin{Bmatrix} \mathbf{R}(t) \\ {}_b\boldsymbol{\omega}(t) \end{Bmatrix} = \begin{Bmatrix} \mathbf{R}(t)_b\boldsymbol{\omega}(t)^\wedge \\ \mathbf{w}(t) \end{Bmatrix}, \tag{3.17}$$

where $\boldsymbol{x}(t) \doteq \{\mathbf{R}(t), {}_b\boldsymbol{\omega}(t)\}$ are the Markov system states. The SDE is nonlinear, so the approach in [69] cannot be leveraged to get exactly sparse linear system. However, similar to approach in [70, 71], it is possible to linearize the system around the current point estimate $\overline{\boldsymbol{x}}(t)$, and achieve a *locally* linear SDE, which can utilize the exactly sparse GP prior proposed in [69].

To define a *locally* linear GP prior, we first look at how to handle uncertainty and define a locally linear GP on SO(3). Various approaches have been proposed to handle uncertainty for SO(3), or even general Lie groups, include [75, 78, 79, 80, 81]. Here the approach in [81] is adopted and define the Gaussian distribution on SO(3) as a Gaussian distribution on the tangent space that is then mapped back to SO(3) by an *exponential map*

$$\widetilde{\mathbf{R}} = \mathbf{R}\exp(\epsilon^{\wedge}), \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \Sigma), \tag{3.18}$$

where $\widetilde{\mathbf{R}}$ is noisy rotation, $\mathbf{R}$ is noise-free rotation, and $\epsilon \in \mathbb{R}^3$ is a small perturbation which is normally distributed with zero mean and Σ covariance.

By adopting this definition of a Gaussian distribution on SO(3) in [81], the GP on SO(3) can be defined locally. Considering a rotation $\mathbf{R}_i$ at time t_i close to $\mathbf{R}(t)$ at t (the time interval between t and t_i is small), the GP model in Eq. (3.1) gives

$$\mathbf{R}(t) = \mathbf{R}_i \exp(\boldsymbol{\xi}_i(t)^{\wedge}), \quad \boldsymbol{\xi}_i(t) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\mathcal{K}}(t_i, t)). \tag{3.19}$$

A *local* variable $\boldsymbol{\xi}_i(t) \in \mathbb{R}^3$ around $\mathbf{R}_i$ is defined as

$$\boldsymbol{\xi}_i(t) \doteq \log(\mathbf{R}_i^{-1}\mathbf{R}(t))^{\vee}, \tag{3.20}$$

where $\vee$ is the inverse operator of $\wedge$, and $\log(\cdot)$ is the *logarithm map* of SO(3), which is

inverse function of the exponential map. The time derivative of $\boldsymbol{\xi}_i(t)$ has [75, p.26]

$$\mathbf{R}(t)^{-1}\dot{\mathbf{R}}(t) = \left(\boldsymbol{\mathcal{J}}_r(\boldsymbol{\xi}_i(t))\dot{\boldsymbol{\xi}}_i(t)\right)^{\wedge}, \tag{3.21}$$

where $\boldsymbol{\mathcal{J}}_r$ is the *right Jacobian* of SO(3) [75, p.40]. With Eq. (3.14) we have

$$\dot{\boldsymbol{\xi}}_i(t) = \boldsymbol{\mathcal{J}}_r(\boldsymbol{\xi}_i(t))^{-1}{}_b\boldsymbol{\omega}(t). \tag{3.22}$$

If the small time interval assumption is satisfied, since $\boldsymbol{\mathcal{J}}_r$ is identity at zero, and $\boldsymbol{\xi}_i(t)$ is close to zero, a good approximation of $\dot{\boldsymbol{\xi}}_i(t)$ is found by

$$\dot{\boldsymbol{\xi}}_i(t) \approx {}_b\boldsymbol{\omega}(t). \tag{3.23}$$

Here $\dot{\boldsymbol{\xi}}_i(t)$ has an explicit meaning: it is the body-frame angular velocity. Considering the case specified by Eq. (3.16) where white noise is injected into time derivative of $\dot{\boldsymbol{\xi}}_i(t)$ by

$$\ddot{\boldsymbol{\xi}}(t) = \mathbf{w}(t), \tag{3.24}$$

the local constant-velocity LTV-SDE of SO(3) is finally written

$$\dot{\boldsymbol{\gamma}}_i(t) = \frac{d}{dt}\begin{bmatrix}\boldsymbol{\xi}_i(t)\\ \dot{\boldsymbol{\xi}}_i(t)\end{bmatrix} = \begin{bmatrix}\dot{\boldsymbol{\xi}}_i(t)\\ \mathbf{w}(t)\end{bmatrix}, \tag{3.25}$$

where $\boldsymbol{\gamma}_i(t) \doteq [\boldsymbol{\xi}_i(t), \dot{\boldsymbol{\xi}}_i(t)]^{\top}$ is the local Markov system states around $\mathbf{R}_i$. The SDE in Eq. (3.25) is linear, so we can apply the approach in Section 3.3.2.

GP Priors on SE(3)

The locally linear constant-velocity GP priors for the *Special Euclidean Group* SE(3) can be defined in similar manner to GP priors on SO(3) in Section 3.3.2. The Special Euclidean

Group SE(3) represents rigid motion in 3D, which is defined by transformation matrices

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{3.26}$$

and where $\mathbf{t}$ is the translation term of motion.

Similar to SO(3), the body-frame velocity ${}_b\mathbf{v}(t)$ has the following relation with the time derivative of the transformation matrix $\dot{\mathbf{T}}(t)$ [77, p.55]

$$\dot{\mathbf{T}}(t) = \mathbf{T}(t){}_b\mathbf{v}(t)^\wedge, \tag{3.27}$$

where $\wedge$ operator constructs a matrix from a velocity $\mathbf{v} \in \mathbb{R}^6$

$$\mathbf{v}^\wedge = \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{bmatrix}^\wedge = \begin{bmatrix} \boldsymbol{\omega}^\wedge & \boldsymbol{v} \\ \mathbf{0} & 0 \end{bmatrix}, \tag{3.28}$$

where $\boldsymbol{v} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}^\top$ is the body-frame translational velocity. The local variable $\boldsymbol{\xi}_i(t)$ is defined by

$$\boldsymbol{\xi}_i(t) \doteq \log(\mathbf{T}_i^{-1}\mathbf{T}(t))^\vee, \tag{3.29}$$

and similar to SO(3), we get the time derivative of local variable $\boldsymbol{\xi}_i(t)$ from Eq. (3.27) $\dot{\boldsymbol{\xi}}_i(t) \approx {}_b\mathbf{v}(t)$. With local variable $\boldsymbol{\xi}_i(t)$ and $\boldsymbol{\gamma}_i(t)$ defined, SE(3) can have locally linear SDE formulated similarly to Eq. (3.25), and an be used to generate a GP prior in the same way.

3.3.3 Sparse GP Priors on Lie Groups

Discussions in Section 3.3.2 and 3.3.2 will be expanded, and a *locally* linear GP prior on general real matrix Lie groups will be formally defined and discussed in this section. We begin by providing notation and several definitions. Every N-dimensional matrix Lie group

G has an associated *Lie algebra* $\mathfrak{g}$ [75, p.16]. The Lie algebra $\mathfrak{g}$ coincides with the local tangent space to the manifold of G. Example Lie algebras of SO(3) and SE(3) are defined by skew symmetric matrices in Eq. (3.15) and Eq. (3.28) respectively. The *exponential map* $\exp : \mathfrak{g} \to G$ and *logarithm map* $\log : G \to \mathfrak{g}$ define the mapping between the Lie group and Lie algebra respectively [75, p.18]. G also has an associating *hat operator* $\wedge : \mathbb{R}^N \to \mathfrak{g}$ and *vee operator* $\vee : \mathfrak{g} \to \mathbb{R}^N$ that convert elements in local coordinates $\mathbb{R}^N$ to the Lie algebra $\mathfrak{g}$ and vice versa [75, p.20].

Constant-Velocity GP Priors on Lie Groups

If $T \in G$ is used to represent an object in G, then the continuous-time trajectory is written as $T(t)$, and trajectory states to be estimated at times $t_1, \ldots, t_M$ are $T_1, \ldots, T_M$. To perform trajectory estimation on G, we first define the Markov system states

$$\boldsymbol{x}(t) \doteq \{T(t), \boldsymbol{\varpi}(t)\}, \tag{3.30}$$

where $\boldsymbol{\varpi}(t)$ is the 'body-frame velocity' variable defined by

$$\boldsymbol{\varpi}(t) \doteq (T(t)^{-1}\dot{T}(t))^{\vee}. \tag{3.31}$$

Since $\forall T \in G,\ T^{-1}\dot{T} \in \mathfrak{g}$ [75, p.20], $\vee$ operator can be applied on $T(t)^{-1}\dot{T}(t)$. In SO(3) and SE(3), $\boldsymbol{\varpi}(t)$ is the body-frame velocity (see Eq. (3.14) and (3.27)). So $\boldsymbol{\varpi}(t)$ is so-called 'body-frame velocity' for general Lie groups. The constant 'body-frame velocity' prior is defined as

$$\dot{\boldsymbol{\varpi}}(t) = \mathbf{w}(t), \quad \mathbf{w}(t) \sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_C\delta(t - t')), \tag{3.32}$$

but this is a nonlinear SDE, which does not match the LTV-SDE defined by Eq. (3.7).

Locally Linear Constant-Velocity GP Priors

To define a LTV-SDE which can leverage the constant-velocity GP prior, we linearize the Lie group manifold around each T_i, and define both a *local* GP and LTV-SDE on the linear tangent space. We first define a local GP for any time t on trajectory which meets $t_i \leq t \leq t_{i+1}$,

$$T(t) = T_i \exp(\boldsymbol{\xi}_i(t)^\wedge), \;\; \boldsymbol{\xi}_i(t) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\mathcal{K}}(t_i, t)). \tag{3.33}$$

the *local* pose variable $\boldsymbol{\xi}_i(t) \in \mathbb{R}^N$ around T_i is defined by

$$\boldsymbol{\xi}_i(t) \doteq \log(T_i^{-1} T(t))^\vee. \tag{3.34}$$

The local LTV-SDE that represents constant-velocity information is

$$\ddot{\boldsymbol{\xi}}_i(t) = \mathbf{w}(t), \;\; \mathbf{w}(t) \sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_C \delta(t - t')). \tag{3.35}$$

If the local Markov system state is defined by

$$\boldsymbol{\gamma}_i(t) \doteq \begin{bmatrix} \boldsymbol{\xi}_i(i) \\ \dot{\boldsymbol{\xi}}_i(t) \end{bmatrix}, \tag{3.36}$$

the local LTV-SDE is rewritten as

$$\dot{\boldsymbol{\gamma}}_i(t) = \frac{d}{dt} \begin{bmatrix} \boldsymbol{\xi}_i(t) \\ \dot{\boldsymbol{\xi}}_i(t) \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\xi}}_i(t) \\ \mathbf{w}(t) \end{bmatrix}. \tag{3.37}$$

To prove the equivalence between the nonlinear SDE in Eq. (3.32) and the local LTV-SDE in Eq. (3.35), we first look at [75, p.26]

$$T(t)^{-1} \dot{T}(t) = \big(\boldsymbol{\mathcal{J}}_r(\boldsymbol{\xi}_i(t)) \dot{\boldsymbol{\xi}}_i(t)\big)^\wedge, \tag{3.38}$$

where $\boldsymbol{\mathcal{J}}_r$ is the *right Jacobian* of G. With Eq. (3.31) we have

$$\dot{\boldsymbol{\xi}}_i(t) = \boldsymbol{\mathcal{J}}_r(\boldsymbol{\xi}_i(t))^{-1}\boldsymbol{\varpi}(t). \tag{3.39}$$

If the small time interval assumption between any t_i and t_{i+1} is satisfied, we have a good approximation of $\dot{\boldsymbol{\xi}}_i(t)$

$$\dot{\boldsymbol{\xi}}_i(t) \approx \boldsymbol{\varpi}(t). \tag{3.40}$$

So we have proved that the LTV-SDE in Eq. (3.35) and (3.37) is a good approximation of constant 'body-frame velocity' prior defined by Eq. (3.32).Note that Section 3.3.2 is a specialization of the above discussion to SO(3).

Both the local GP and LTV-SDE are defined on the tangent space, so they are only valid around current linearization points T_i. But if all time stamps have a small enough interval, the GP and LTV-SDE can be defined in a piecewise manner, and every point on the trajectory can be converted to local variable $\boldsymbol{\xi}_i(t)$ based on its nearby estimated state T_i.

A Factor Graph Perspective

Once the local GP and constant-velocity LTV-SDE are defined, the cost function J_{gp} used to incorporate information about the GP prior into the nonlinear least squares optimization can be written down in Eq. (3.3). As discussed, the GP prior cost function has the generic form

$$J_{gp} = \frac{1}{2} \parallel \boldsymbol{\mu} - \boldsymbol{x} \parallel^2_{\boldsymbol{\mathcal{K}}}, \tag{3.41}$$

but if the trajectory is generated by a constant-velocity LTV-SDE in Eq. (3.37), the GP prior cost can be specified as as [69]

$$J_{gp} = \sum_i \frac{1}{2}\mathbf{e}_i^\top \mathbf{Q}_i^{-1}\mathbf{e}_i, \tag{3.42}$$

$$\mathbf{e}_i = \boldsymbol{\Phi}(t_{i+1}, t_i)\boldsymbol{\gamma}_i(t_i) - \boldsymbol{\gamma}_i(t_{i+1}), \tag{3.43}$$

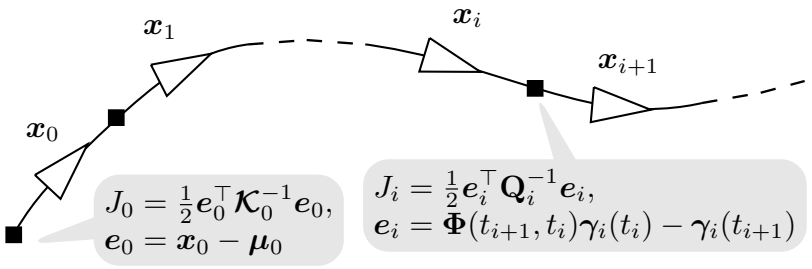


Figure 3.1: An example factor graph, showing states (triangles) and factors (black boxes). GP prior factors connect consecutive states, and define the prior information on first state.

where $\mathbf{Q}_i$ is covariance matrix by [69]

$$\boldsymbol{\Phi}(t,s) = \begin{bmatrix} \mathbf{1} & (t-s)\mathbf{1} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}, \mathbf{Q}_i = \begin{bmatrix} \frac{1}{3}\Delta t_i^3 \mathbf{Q}_C & \frac{1}{2}\Delta t_i^2 \mathbf{Q}_C \\ \frac{1}{2}\Delta t_i^2 \mathbf{Q}_C & \Delta t_i \mathbf{Q}_C \end{bmatrix}, \tag{3.44}$$

where $\Delta t_i = t_{i+1} - t_i$.

Since the GP prior cost J_{gp} has been written as a sum of squared cost terms, and each cost term is only related to nearby (local) Markov states, the least square problem is represented by *factor graph* models. In factor graphs the system states are represented by variable factors, and the cost terms are represented by cost factors. An example factor graph is shown in Fig. 3.1. By converting nonlinear least squares problems into factor graphs we can take advantage of factor graph inference tools to solve the problems efficiently. Additional information about the relationship between factor graphs and sparse GP and SLAM problems can be found in [53, 69, 70, 74]

Querying the Trajectory

One of the advantages of representing the continuous-time trajectory as a GP is that we have the ability to query the state of the robot at any time along the trajectory. For constant-velocity GP priors, the system state $\boldsymbol{x}(\tau), t_i \leq \tau \leq t_{i+1}$ can be estimated by two nearby states $\boldsymbol{x}(t_i)$ and $\boldsymbol{x}(t_{i+1})$ [69], which allows efficient $O(1)$ interpolation. The mean value of

local state $\hat{\boldsymbol{\gamma}}_i(\tau)$ is first calculated by

$$\hat{\boldsymbol{\gamma}}_i(\tau) = \boldsymbol{\Lambda}(\tau)\hat{\boldsymbol{\gamma}}_i(t_i) + \boldsymbol{\Psi}(\tau)\hat{\boldsymbol{\gamma}}_i(t_{i+1}), \tag{3.45}$$

where

$$\boldsymbol{\Lambda}(\tau) = \boldsymbol{\Phi}(\tau, t_i) - \mathbf{Q}_\tau \boldsymbol{\Phi}(\tau, t_i)^\top \mathbf{Q}_{i+1}^{-1} \boldsymbol{\Phi}(t_{i+1}, t_i), \tag{3.46}$$

$$\boldsymbol{\Psi}(\tau) = \mathbf{Q}_\tau \boldsymbol{\Phi}(\tau, t_i)^\top \mathbf{Q}_{i+1}^{-1}. \tag{3.47}$$

Once the mean value of local state $\hat{\boldsymbol{\gamma}}_i(\tau)$ is decided, the mean value of the full state $\hat{\boldsymbol{x}}(\tau) = \{\hat{T}(\tau), \hat{\boldsymbol{\varpi}}(\tau)\}$ is

$$\hat{T}(\tau) = \hat{T}_i \exp\Big(\big(\boldsymbol{\Lambda}_1(\tau)\hat{\boldsymbol{\gamma}}_i(t_i) + \boldsymbol{\Psi}_1(\tau)\hat{\boldsymbol{\gamma}}_i(t_{i+1})\big)^\wedge\Big), \tag{3.48}$$

$$\hat{\boldsymbol{\varpi}}(\tau) = \boldsymbol{\mathcal{J}}_r\big(\hat{\boldsymbol{\xi}}_i(\tau)\big)^{-1}\big(\boldsymbol{\Lambda}_2(\tau)\hat{\boldsymbol{\gamma}}_i(t_i) + \boldsymbol{\Psi}_2(\tau)\hat{\boldsymbol{\gamma}}_i(t_{i+1})\big), \tag{3.49}$$

where

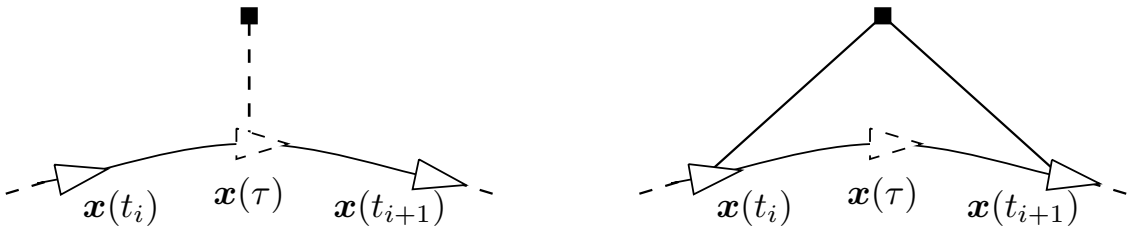
$$\boldsymbol{\Lambda}(\tau) = \begin{bmatrix} \boldsymbol{\Lambda}_1(\tau) \\ \boldsymbol{\Lambda}_2(\tau) \end{bmatrix}, \ \boldsymbol{\Psi}(\tau) = \begin{bmatrix} \boldsymbol{\Psi}_1(\tau) \\ \boldsymbol{\Psi}_2(\tau) \end{bmatrix},$$

$$\hat{\boldsymbol{\gamma}}_i(t_i) = \begin{bmatrix} \mathbf{0} \\ \hat{\boldsymbol{\varpi}}(t_i) \end{bmatrix}, \ \hat{\boldsymbol{\gamma}}_i(t_{i+1}) = \begin{bmatrix} \hat{\boldsymbol{\xi}}_i(t_{i+1}) \\ \boldsymbol{\mathcal{J}}_r(\hat{\boldsymbol{\xi}}_i(t_{i+1}))^{-1}\hat{\boldsymbol{\varpi}}(t_{i+1}) \end{bmatrix},$$

$$\hat{\boldsymbol{\xi}}_i(\tau) = \log(\hat{T}_i^{-1}\hat{T}(\tau))^\vee, \quad \hat{\boldsymbol{\xi}}_i(t_{i+1}) = \log(\hat{T}_i^{-1}\hat{T}_{i+1})^\vee,$$

Fusion of Asynchronous Measurements

The real power of continuous-time trajectory interpolation is not limited to obtaining the system state at any time of interest, but it also provides the ability to incorporate measurements in MAP estimation in Eq. (3.3) not only at estimated time stamps t_i, but also at any time. This provides a framework of sensor fusion when asynchronous measurements at



(a) Meausurement (b) Interpolated Factor

Figure 3.2: (a) Measurement at time τ, dashed line indicates it's not an actual factor. (b) The interpolated factor encodes measurement at time τ.

present, also enabling us to reduce the number of states optimized by skipping some states without losing any measurement information.

Assume there is a measurement $\mathbf{z}_\tau$ of state $\boldsymbol{x}(\tau)$ available at an arbitrary time $\tau, t_i \leq \tau \leq t_{i+1}$, with measurement function $\mathbf{h}_\tau$ and corresponding covariance $\boldsymbol{\Sigma}_\tau$. The measurement cost in the factor graph can be written as

$$J_\tau(\boldsymbol{x}(\tau)) = \frac{1}{2} \parallel \mathbf{z}_\tau - \mathbf{h}_\tau(\boldsymbol{x}(\tau)) \parallel^2_{\boldsymbol{\Sigma}_\tau} . \tag{3.50}$$

Since the system state $\boldsymbol{x}(\tau)$ is not explicitly available during optimization, a trajectory interpolation is performed between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ by Eq. (3.48) – (3.49), and rewrite the cost in terms of the interpolated mean value $\hat{\boldsymbol{x}}(\tau)$

$$J_\tau(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}) = \frac{1}{2} \parallel \mathbf{z}_\tau - \mathbf{h}_\tau(\hat{\boldsymbol{x}}(\tau)) \parallel^2_{\boldsymbol{\Sigma}_\tau} . \tag{3.51}$$

Because the measurement cost is represented by $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$, a binary factor can be added to the factor graph and optimized without explicitly adding an additional state. An example is illustrated in Fig. 3.2.

Simultaneous Trajectory Estimation and Mapping

The proposed approach could be extended from trajectory estimation to simultaneous trajectory estimation and mapping (STEAM) [69] by combining landmarks $\boldsymbol{l}$ with trajectory

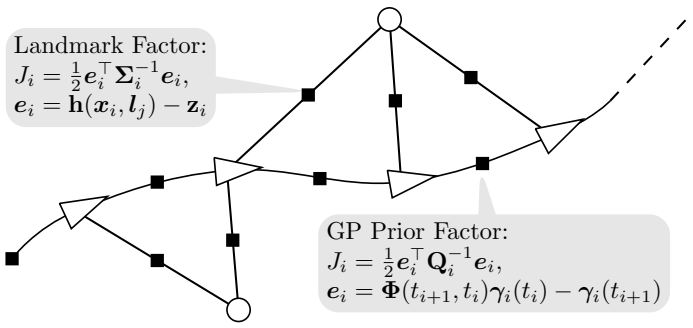


Figure 3.3: A factor graph of an example STEAM problem containing GP prior factors and landmark measurements factors. Landmarks are illustrated with open circles.

states $\boldsymbol{x}$, resulting a combined state $\boldsymbol{z} \doteq [\boldsymbol{x}, \boldsymbol{l}]^\top, \boldsymbol{l} \doteq [\boldsymbol{l}_1, \ldots, \boldsymbol{l}_L]^\top$, where L is number of landmarks [69]. The resulting linear system has the form

$$\underbrace{\begin{bmatrix} \mathbf{W}_{xx} & \mathbf{W}_{lx}^\top \\ \mathbf{W}_{lx} & \mathbf{W}_{ll} \end{bmatrix}}_{\mathbf{W}} \underbrace{\begin{bmatrix} \delta\boldsymbol{x}^* \\ \delta\boldsymbol{l}^* \end{bmatrix}}_{\delta\boldsymbol{z}^*} = \underbrace{\begin{bmatrix} \boldsymbol{b}_x \\ \boldsymbol{b}_l \end{bmatrix}}_{\boldsymbol{b}}, \tag{3.52}$$

where $\mathbf{W}_{xx}$ is block-tridiagonal due to the GP prior, $\mathbf{W}_{ll}$ is block-diagonal, and $\mathbf{W}_{lx}$ depends on landmark observations, but is generally sparse [53]. Block-wise sparse Cholesky decomposition can be applied to solve the linear system

$$\underbrace{\begin{bmatrix} \mathbf{V}_{xx} & \mathbf{0} \\ \mathbf{V}_{lx} & \mathbf{V}_{ll} \end{bmatrix}}_{\mathbf{V}} \underbrace{\begin{bmatrix} \mathbf{V}_{xx}^\top & \mathbf{V}_{lx}^\top \\ \mathbf{0} & \mathbf{V}_{ll}^\top \end{bmatrix}}_{\mathbf{V}^\top} = \underbrace{\begin{bmatrix} \mathbf{W}_{xx} & \mathbf{W}_{lx}^\top \\ \mathbf{W}_{lx} & \mathbf{W}_{ll} \end{bmatrix}}_{\mathbf{W}}. \tag{3.53}$$

A factor graph representing an example STEAM problem is shown in Fig. 3.3.

3.3.4 Evaluation

The proposed GP framework is evaluated on three different trajectory estimation tasks on SE(2), SO(3) and Sim(3). Trajectory estimation on SE(3) (a special case of our general Lie group formulation) has been reported in [71], so we do not repeat those evaluations.

Table 3.1: Planar SLAM comparison, Linear and SE(2).

	Plaza1		**Plaza2**	
	Linear	SE(2)	Linear	SE(2)
Position RMS (m)	0.252	0.238	0.523	0.152
Rotation RMS (deg)	2.822	2.508	1.952	0.981
Landmark RMS (m)	0.053	0.026	0.479	0.029
Optimization Time (s)	1.998	2.107	0.832	0.888

SE(2): 2D Planar Range-Only SLAM

The experiments of SE(2) are conducted on two range-only 2D SLAM problems to evaluate the proposed GP approach on SE(2). The datasets are *Plaza1* and *Plaza2* from Djugash *et al*. [82]. Both datasets were collected by a lawn mower equipped with a gyroscope and wheel encoders to measure odometry, and a radio node measuring ranges of four fixed beacons. Ground truth beacon positions and robot trajectories were measured by RTK-GPS. Fig. 3.4 shows the results, including ground-truth trajectories and landmark positions, and odometry-only dead reckoning trajectories. Trajectory error and trajectory distribution (shown by 3σ variance) estimated for Plaza1 dataset are plotted in Fig. 3.5. In the middle of the Plaza1 dataset, the robot loses range measurements to all beacons. From the estimated trajectory distribution we see the uncertainty increases in response to this missing data.

The performance of the proposed GP approach on SE(2) is compared with the naïve sparse linear GP prior [69], which uses canonical coordinates $\boldsymbol{x}(t) = [x(t), y(t), \theta(t)]^\top \in \mathbb{R}^3$ as system state. The accuracy and efficiency of both approaches are evaluated by root mean square (RMS) error and optimization time respectively. Table 3.1 shows the comparison results. Both trajectory and landmark estimation accuracies are improved by our Lie group approach. Since both of approaches share the same sparse factor graph representation, there is no significant difference of efficiency between these two approaches. Optimization timing results support above argument, since there is only less than 10% computational time difference between both datasets.

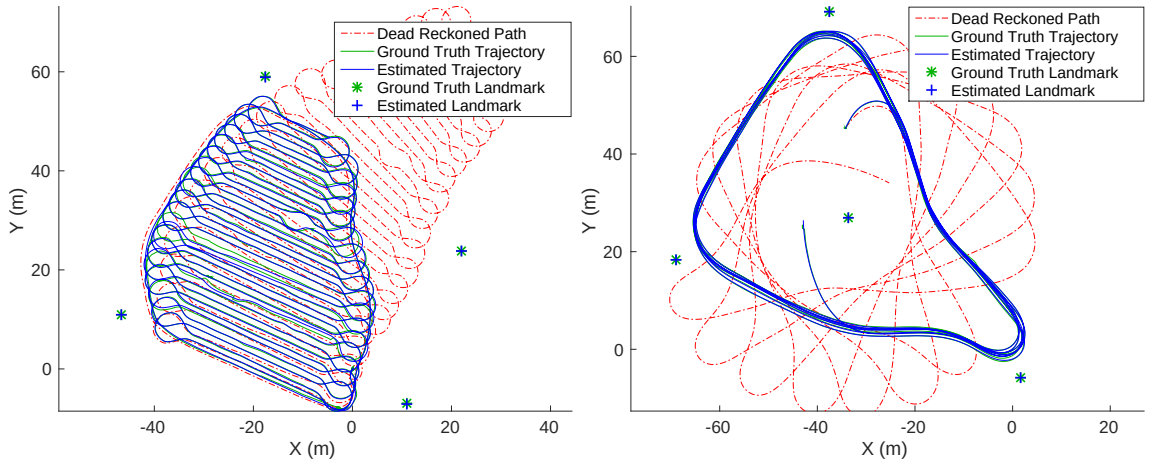


Figure 3.4: Planar SLAM results on the Plaza1/2 datasets [82], with odometry-only and ground truth trajectories.

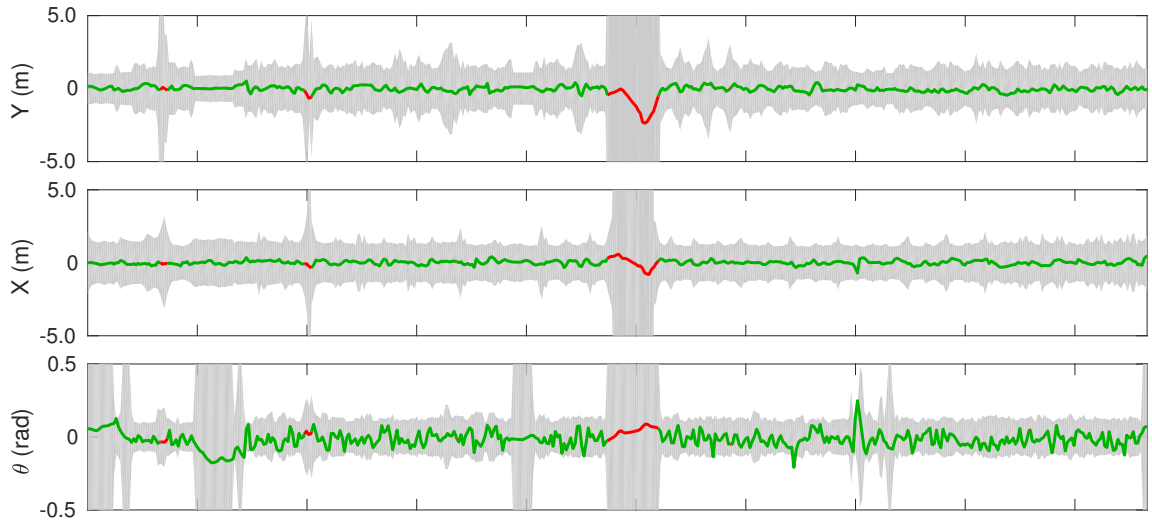


Figure 3.5: Trajectory error and 3σ variance estimated of Plaza1 dataset. Green lines are states with range measurements, and red segments are states without range measurements.

SO(3): 3D Attitude Estimation of IMU

To evaluate our GP approach on SO(3), an attitude estimation experiment is designed and performed using an inertial measurement unit (IMU). An IMU dataset is collected using a low-cost Pixhawk autopilot, which has a 3-axis accelerometer and a 3-axis gyroscope. Both acceleration and angular rates were collected asynchronously, as angular rate was available at 166Hz but acceleration was available at 40Hz. Ground truth attitude was collected by an Optitrack motion capture system.

A major advantage of using continuous-time trajectory representations is that estimation with asynchronous sensors can be accomplished simply. A batch attitude estimation approach is implemented with pre-integrated IMU factors [81] containing gyroscope measurements, and GP interpolated acceleration factors which compare acceleration measurements against gravity. After optimization, attitudes are estimated at the gyroscope measurement time stamps. Since the accelerometer does not provide yaw angle information with respect to the world frame, only pitch and roll angles are estimated and compared with ground truth.

Estimated pitch and roll angle errors from the IMU dataset are shown in Fig. 3.6, compared with gyroscope-only estimation, and compared with motion capture ground truth. The results show that both estimated pitch and roll angles are better aligned with ground truth, since the gyroscope drift from bias is compensated for by fusing asynchronous acceleration measurements and sensor bias is also estimated. The existence of the few peaks in roll angle error plot are due to singularity caused by the Euler angle representation.

Sim(3): Scale Drift Aware Monocular SLAM

Although transformations of 3D rigid bodies are represented by elements of SE(3), in monocular visual SLAM, trajectories suffer scale drift if they are represented on SE(3), due to lack of scene scale information. In monocular visual SLAM, camera motion and scene structure are recovered up to scale, due to the projective nature of a single camera. Although the scale of monocular visual SLAM is locally consistent, the scale estimate suffers from drift due to the lack of an anchor. Several approaches [83, 54] have been proposed to solve this issue by estimating the trajectory on Similarity group Sim(3) [83], which is defined by

$$\mathbf{S} = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \mathbf{R} \in \mathrm{SO}(3), \tag{3.54}$$

where s is the estimated local scale.

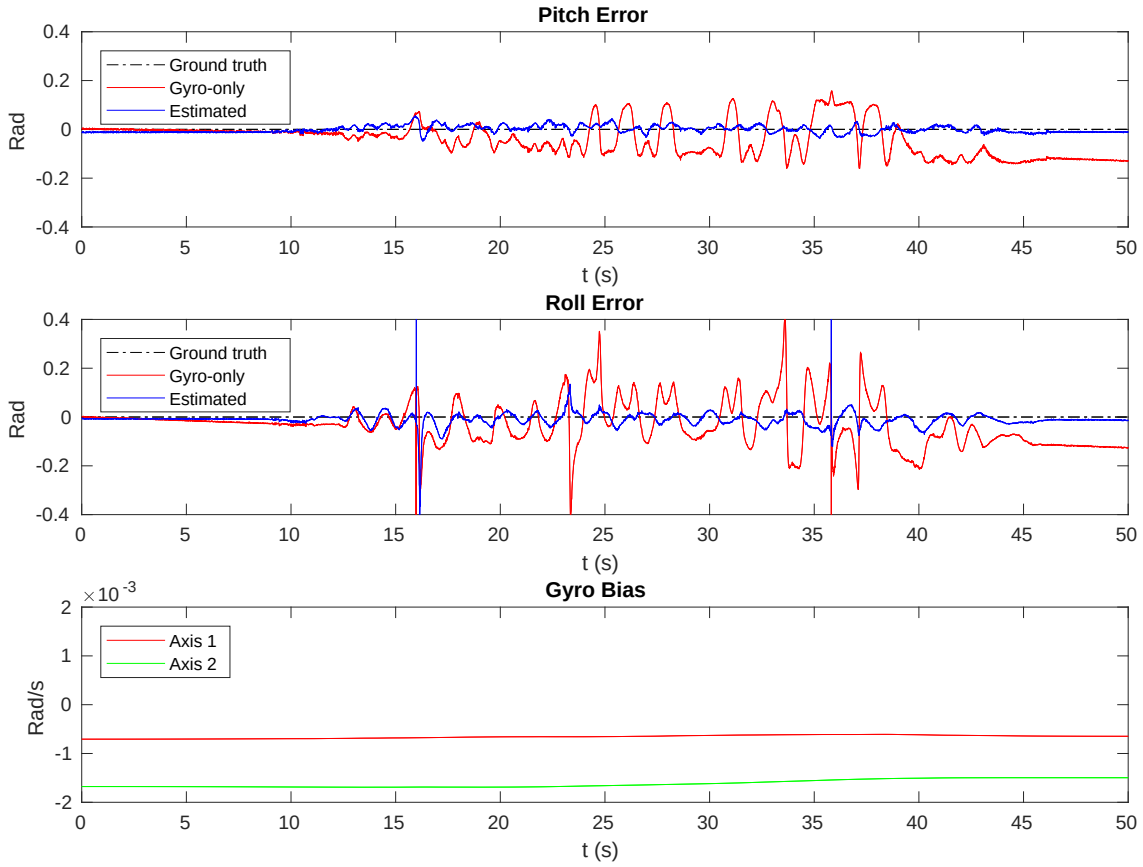


Figure 3.6: Attitude estimation errors by proposed approach of IMU dataset, compared with gyroscope-only results, and estimated gyroscope bias by proposed approach.

Sparse GPs are implemented on Sim(3), and monocular visual SLAM experiments are conducted to show how GPs help with estimation on Sim(3). A hand-held monocular camera with an IMU is built, and it's used to collected a forward looking outdoor dataset as shown in Fig. 3.7(a). The camera and IMU readings are also collected asynchronously, similar to the IMU dataset. A visual odometry (VO) algorithm which is similar to what is described in Section 3.4 is used to estimate camera trajectories, but without IMU and GPS measurements. The VO result of a Georgia Tech campus dataset is illustrated in Fig. 3.7(b). The scale drift is clearly shown, since the path is no longer a closed loop. A loop closure measurement is received near the trajectory end, marked in red (in Fig. 3.7(b)) which includes relative scale information [83].

I implemented and tested three methods for comparison: a 6-DOF SE(3) pose graph, a

7-DOF Sim(3) pose graph [83], and a 7-DOF Sim(3) pose graph with GP and IMU factors. All results are shown in Fig. 3.7. Compared to the ground truth map, the estimated 6-DOF pose graph is distorted due to scale drift; the 7-DOF Sim(3) pose graph is better but still distorted; and the proposed 7-DOF Sim(3) on GP with IMU approach achieves the best result.

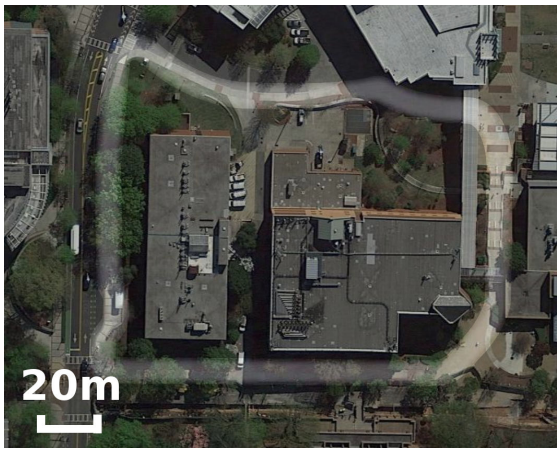
To understand why the 7-DOF approach with GPs outperforms the one without GPs, the estimated local scales for both approaches are plotted in Fig. 3.7(f). Although the loop closure gives relative scale information at the start and end of the estimation, there is no other information given in the middle of the trajectory, so the 7-DOF Sim(3) pose graph assumes the scale drift changes at a near-constant rate during the whole length of the trajectory. But that's not the case in the dataset. With the help from GP interpolation, asynchronous IMU measurements provide more information about how relative scale drifts in the middle of the trajectory, and as shown in Fig. 3.7(f) that our Sim(3) approach, which uses GPs and IMU, gives scale estimation of non-constant changing rate, leading to better SLAM results.

3.4 Multi-sensor SLAM for 3D reconstruction

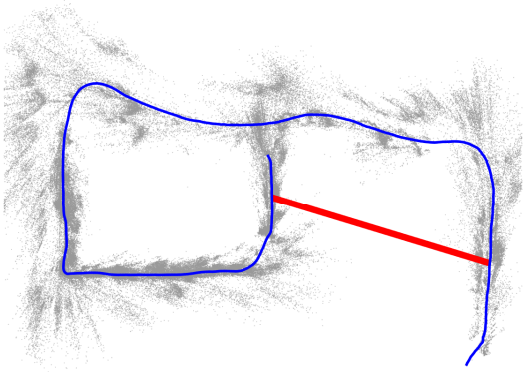
3.4.1 Technical approach

In this section I propose a multi-sensor SLAM pipeline for 3D reconstruction which has two parts, the front-end module and the back-end module, as shown in Figure 3.8. The front-end module processes images for visual landmarks, and the back-end module estimates camera states and landmarks using visual landmark information from the front-end and other sensor inputs.

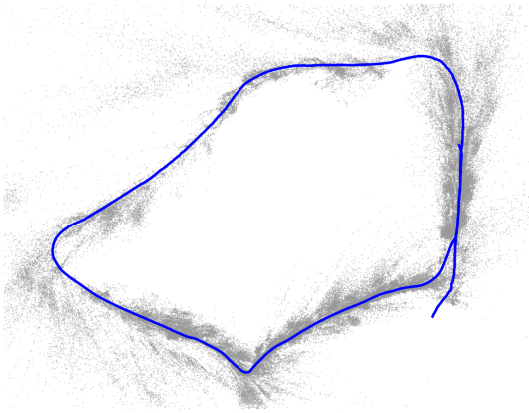
The front-end module processes the input images to landmark information, which can be further utilized in the factor graph that underlies the sensor fusion process. The input image frames are processed by the following protocol: First the images are undistorted to remove any radial distortion generated by the wide FOV camera lens. Second



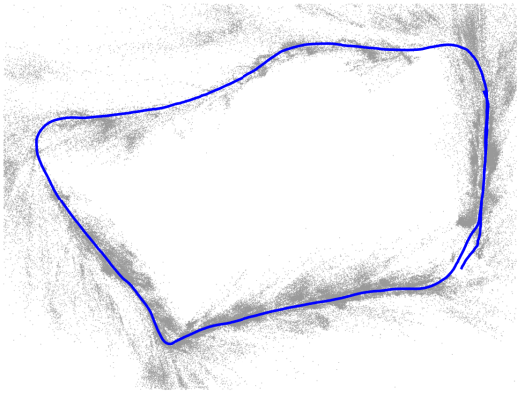
(a) Google Map view



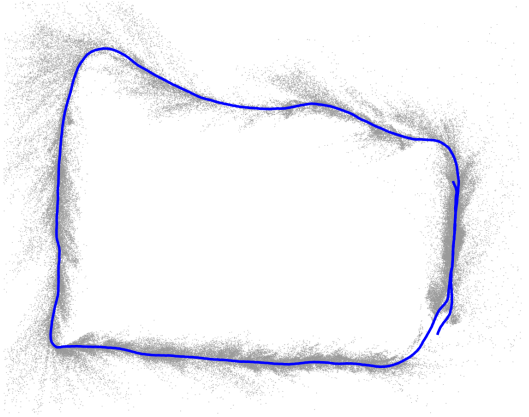
(b) Open loop VO



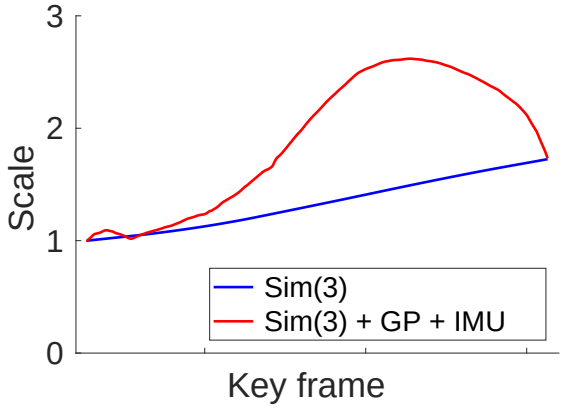
(c) SE(3) pose graph



(d) Sim(3) pose graph



(e) Sim(3) with GP and IMU



(f) Estimated scales

Figure 3.7: Scale drift aware monocular SLAM results. (a) Google Map view of the map with path highlighted; (b) shows open loop VO results, with the loop closure marked in red; (c)-(e) are SE(3) and Sim(3) results, and (f) shows scale estimations of Sim(3) approaches.

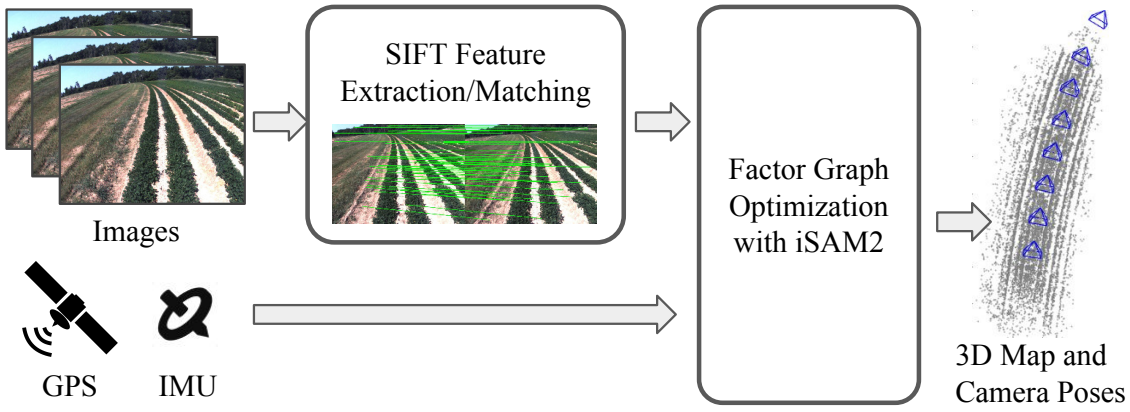


Figure 3.8: Overview of multi-sensor SLAM system.

SIFT [84] features are extracted on each undistorted image. Third SIFT descriptors are matched in nearby image pairs by following protocol: image pairs are selected when two image frames are nearby in time stamps (time interval between time stamp smaller than 1 second), then SIFT descriptors in selected pairs are matched by the approximate nearest neighbor library FLANN [85], finally matches in each image pair are further filtered by 8-point RANSAC [86] (or equivalently 5-point RANSAC [87], since the camera calibration is known) to reject outliers. Finally a single visual landmark is accepted if there are more than 6 frames that have corresponding features matched to the same landmark.

The back-end module of the 3D reconstruction pipeline uses visual landmark information from the front-end module, plus the sensor information from other sensors, and performs a multi-sensor SLAM by GP regression on SE(3) proposed in previous section. Since the goal of the multi-sensor SLAM system is to reconstruct a *single* row during a *single* data collection session, here we first define the states estimates by the back-end module: the SLAM problem here with N images input and various other sensor information, the task is to estimate a set of N camera states $X^{\langle t_i,r_j\rangle} = \{\boldsymbol{x}_0^{\langle t_i,r_j\rangle}, ..., \boldsymbol{x}_{N-1}^{\langle t_i,r_j\rangle}\}$ at row r_j and session t_i when the dataset are taken, given visual landmark measurements from the front-end module, and other sensor measurements, including an Inertial Measurement Unit (IMU) and GPS.

The GP regression problem is formulate by a factor graph [53], and solved by per-

forming a Maximum a Posteriori (MAP) estimation. For this section if not mentioned separately, we use i to index different dataset sessions, j to index different rows in a single session, k and l to index different camera states and sensor measurements in a single session an a single row respectively. Each camera state to be estimated is defined by a set $\boldsymbol{x}_k = \{\mathbf{T}_k, \boldsymbol{v}_k, \boldsymbol{\omega}_k, \mathbf{b}_k\}$, which includes camera pose $\mathbf{T}_k \in$ SE(3), translational velocity $\boldsymbol{v}_k \in \mathbb{R}^3$, angular rotation rate $\boldsymbol{\omega}_k \in \mathbb{R}^3$, and IMU sensor bias $\mathbf{b}_k \in \mathbb{R}^3$. The SLAM problem is formulated as a GP regression problem, on a factor graph where the joint probability distribution of set of all estimated variables $X^{\langle t_i, r_j \rangle} \doteq \{\boldsymbol{x}_k\}$ given all sensor measurements $Z^{\langle t_i, r_j \rangle}$ is factorized and represented as the product

$$p(X^{\langle t_i, r_j \rangle} | Z^{\langle t_i, r_j \rangle}) \propto \phi_{GP}(X^{\langle t_i, r_j \rangle}) \prod_{l=1}^{L} \phi(X_l^{\langle t_i, r_j \rangle}), \tag{3.55}$$

where $\phi_{GP}(X^{\langle t_i, r_j \rangle})$ is the GP prior on all states $X^{\langle t_i, r_j \rangle}$, L is the total number of factors, $X_l^{\langle t_i, r_j \rangle}$ is the set of variables the lth factor involved, and ϕ is the factor in the graph which is proportional to measurement likelihood $l(X_k^{\langle t_i, r_j \rangle}; z_l^{\langle t_i, r_j \rangle})$, given lth measurement $z_k^{\langle t_i, r_j \rangle} \in Z^{\langle t_i, r_j \rangle}$. The states can then be computed by MAP estimation

$$\hat{X}^{\langle t_i, r_j \rangle} = \underset{X^{\langle t_i, r_j \rangle}}{\operatorname{argmax}}\, p(X^{\langle t_i, r_j \rangle} | Z^{\langle t_i, r_j \rangle}). \tag{3.56}$$

Various types of factors are used in this multi-sensor SLAM pipeline, and an example factor graph is shown in Fig. 3.9. Structure-less smart factors [88] are used for visual landmarks, to reduce memory storage by avoiding the explicit estimation of landmark variables. Outlier rejection of vision factor is enabled, which behaves similar to M-Estimators [89], to avoid wrong landmarks generated by miss matching of descriptor matching at frond-end. IMU measurements are incorporated into the factor graph by pre-integrated IMU factors [81]. GPS measurements are queried at time stamps not synchronized with camera time stamp (see Sec. 2), so interpolated binary factors (magenta factors in Fig. 3.9) are used to incorporated these asynchronous measurements. Finally GP prior factors are shown in

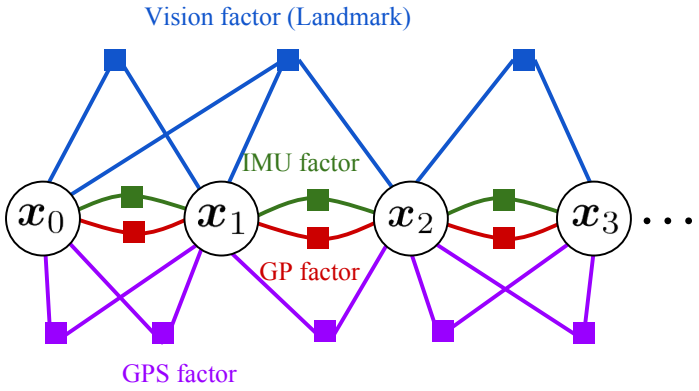


Figure 3.9: Factor graph of multi-sensor SLAM, with system states represented by circles, and various factors represented by colored squares: structure-less landmark factors, IMU factors, GP prior factors and GP-interpolated GPS factors.

red in Fig. 3.9).

The factor graph is optimized by iSAM2 [90]. Once camera states are estimated, M landmarks $L^{\langle t_i, r_j \rangle} = \{l_0^{\langle t_i, r_j \rangle}, ..., l_{M-1}^{\langle t_i, r_j \rangle}\}$ are triangulated by known camera poses.

3.4.2 Evaluation

Evaluation of the proposed multi-sensor SLAM method is performed on the ground vehicle dataset collected, and results are shown in Figure 3.10 and Figure 3.11. Figure 3.10 shows the estimated trajectory of camera/IMU/GPS input with ground truth trajectory from RTK-GPS. The full trajectory lasts about 40 minutes and contains about 15k frames, proposed multi-sensor SLAM gets the result using less than 16GB memory, this shows proposed multi-sensor SLAM runs on large-scale and long-term datasets with reasonable memory usage. The drift shown in the results are caused by long-term drift caused by GPS measurements, and not easy to be fixed with other sensor data (like visual landmarks and IMU) in current setting. We will discuss how this issue can be solved in next section, by running cross-row feature matching. Figure 3.10 shows example projected colorized landmarks, triangulated by smart factors.

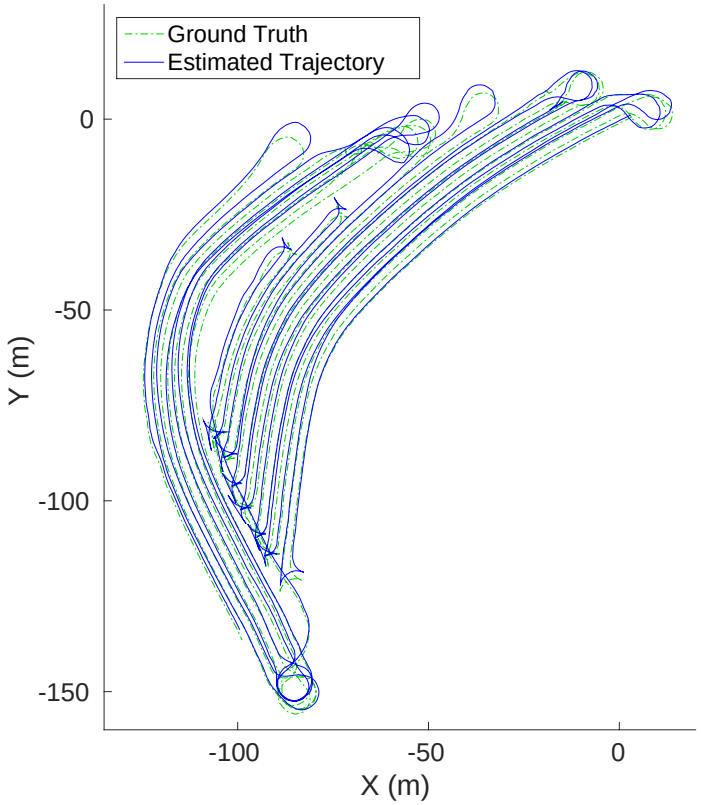


Figure 3.10: An estimated trajectory of multi-sensor SLAM system on 2016 field dataset.

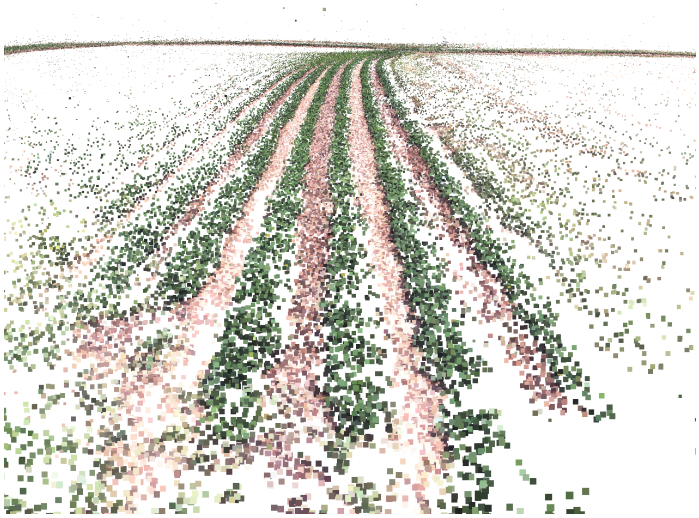


Figure 3.11: Triangulated landmarks of multi-sensor SLAM system on 2016 field dataset.

CHAPTER 4

SPATIO-TEMPORAL (4D) RECONSTRUCTION

4.1 Introduction and claims

We can obtain 3D reconstructions of fields by the proposed multi-sensor SLAM method, however, we cannot handle changing scenes in which the crops are growing continuously, which needs to be handled in precision agriculture applications. A single 3D reconstruction provides us 3D information of crops (e.g. height and canopy size) at current time, but cannot provide us any *temporal* 3D information of crops, such as growth rate and color changes. People may argue such information is available with multiple 3D reconstructions. But if only multiple 3D reconstructions are available, we will not be able to get any temporal information of a single location or a single plant, since we cannot build correspondence between 3D reconstructions.

To solve the above issue, we consider the idea of *spatio-temporal* reconstruction, also called *4D reconstruction* (3D + time) in this chapter. The goal of 4D reconstruction is to build a sequence of 3D reconstructions into a single coordinate frame. During the whole growing season, the field dataset is collected periodically, and each session lasts no more than a few hours. The crops won't grow significantly in a few hours, so it is safe to assume that the field is *static* and run 3D reconstruction in a single session. The idea of 4D reconstruction is shown in Fig. 4.1, which shows a 4D reconstruction consists of many 3D reconstructions.

Since 3D reconstructions of each session were already performed in their own coordinate frames, the 4D reconstruction problem now comes down to solving a registration of a set of 3D reconstructions into *a single* coordinate frame. Fig. 4.2 shows how a 4D reconstruction is performed between two 3D reconstructions. In the figure two 3D recon-

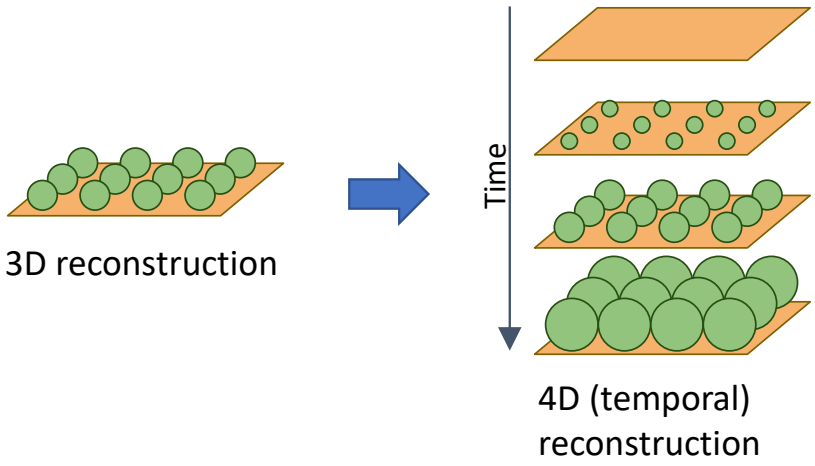


Figure 4.1: Idea of exteding 3D reconstruction to temporal 3D (4D) reconstruction, a multi-row 3D reconstruction is extended to a multi-session and multi-row 4D reconstruction.

structions captured at different times are shown in red and blue respectively, and triangle cones and circles represent cameras and landmarks respectively. The 4D reconstruction is shown on the right, that both 3D reconstructions are aligned into session 1's coordinate frame (session 2's frame is also OK). In the proposed 4D reconstruction pipeline, since the field has multi-row structure (as shown in Fig. 4.1), 3D reconstructions are done in single-row manner, so cross-row alignments are also needed.

Although GPS sensor data was collected alongside the dataset, which registers all 3D reconstructions into the world coordinate frame, there are significant GPS errors that prevent us from obtaining accurate registrations between different sessions. Therefore we need to find a better way than sole GPS to align 3D reconstructions into a single frame. The solution of registration problem in 3D we used is to build visual data association (visual correspondences) between multiple 3D reconstructions. The input data association selected for the 4D reconstruction pipeline is visual correspondences between 2D images. Visual correspondences between 2D images can provide data association in 3D. If two 2D image points are matched in a image pair, they should belong to the same 3D object, and the two corresponding 3D points (in their original two 3D reconstructions) should be the same 3D point in final 4D reconstruction. Given such visual correspondences information, multiple 3D reconstructions can be aligned into a single frame, close the loop of 4D

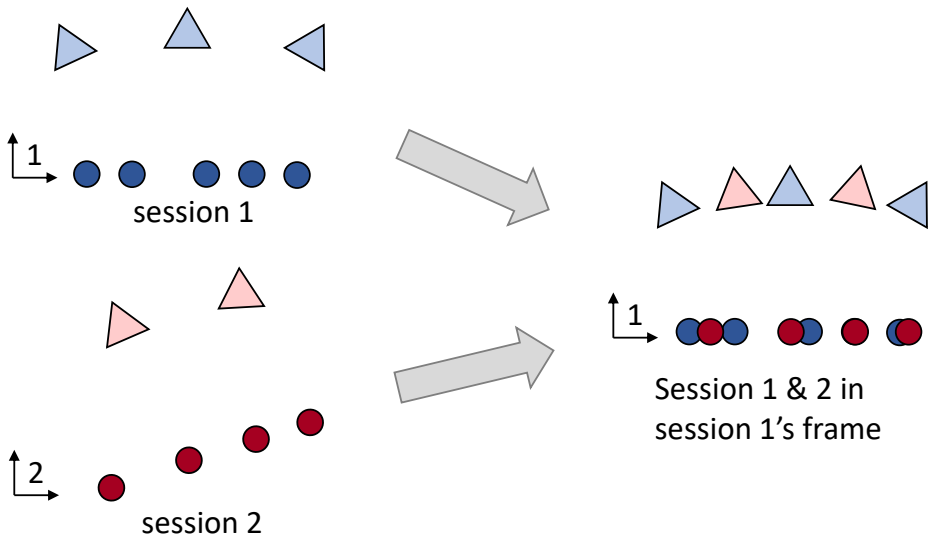


Figure 4.2: Aligning two 3D reconstrucions into a single coordinate frame. Two 3D reconstructions are shown in red and blue on left, and on the right we show the aligned 4D reconstruction in blue's frame. See text.

reconstruction.

In this chapter, I propose a full pipeline for performing 4D reconstruction given multiple 3D reconstructions from multiple sessions, which are captured at different times of a growing season. My proposed 4D reconstruction pipeline has two parts:

1. A data association method for matching images over possibly time and large baselines. The data association method provides visual correspondence information for 4D reconstruction. In the context of this thesis, the term "data association" has the same meaning of "feature matching", a.k.a. looking for "visual correspondence".

2. A factor graph based 4D reconstruction method based on the general GP-based continuous-time SLAM framework. The factor graph based 4D reconstruction is an extension of 3D reconstruction pipeline proposed in last chapter, which jointly optimizes multiple 3D reconstructions in a single coordinate frame by accounting visual correspondence information.

My claims for the proposed 4D reconstruction pipeline are:

- *Low cost*: the proposed 4D reconstruction pipeline works for low-cost precision agriculture systems, by using monocular camera input and avoiding expensive LIDAR.
- *Accurate*: the proposed 4D reconstruction pipeline gives 4D reconstruction results as accuracy in 2cm, which is enough for most precision agriculture applications.
- *Flexible*: the proposed 4D reconstruction pipeline is plug-and-play to any type of sensor, and no hardware synchronization needed, since it is based on the previously proposed general GP-based continues-time SLAM framework.

This chapter summarizes and extends materials from my previous publication [47].

4.2 Related work

Change detection Change detection has been studied on various geometric entities, including 2D images [91, 92], 3D point clouds [93], and 3D volumes [94, 31]. Taneja et al. [95] propose a 3D reconstruction of changed parts from 2D images, and has applied the approach to city-scale change detection and 3D reconstruction [96, 97]. Pollard et al. [94] utilize 3D volume information helps change detection on 2D images and improve performance. Martin et al. [98, 99] synthesize smooth time-lapse videos from images collected on the Internet. But this work is limited to 2D video results, although 3D depth information of the scene is used while synthesizing the videos.

2D and 3D change detection has been used in applications like urban construction site monitoring [100] and time-lapse video synthesizing [98, 99]. However, scene change detection by itself is not enough for crop monitoring applications, since it focuses on appearance changes, and does not apply to precision agriculture applications which need metric dimensional information of crops, where crops are *continuously* changing over time.

4D reconstruction 4D reconstruction is not a new idea, and has been studied in recent years. Early work in 4D reconstruction include [27, 28], which build city-scale 3D re-

constructions via temporal inference on historical images. Schindler et al. first formulate the temporal order inference problems of 3D structures as constraint satisfaction problems (CSPs), and solve CSPs on images [27], then the work is extended to a full probabilistic temporal inference solved on 3D [28]. Recent works of 4D reconstruction include [29], in which the authors assume the environment can be partitioned into multiple planar segmentations, each one has constant appearance for a period of time. A probabilistic volumetric 4D representation of the environment was proposed by [30], in [30] octree volumetric representations of 3D reconstruction are used, and a full probabilistic framework is proposed to estimate appearance and surface of the 4D model. The major issue with most existing approaches is that they assume each geometric entity keeps nearly-constant appearance for the temporal duration, which is not the case in crop monitoring since crops are changing *continuously*.

Deformable SfM Another topic which is closely related to 4D reconstruction is *Deformable 3D reconstruction*, which assumes the geometric strictures to be reconstructed are deformable. Most existing deformable SfM works are extensions on factorization based SfM method [50]. The basic deformable SfM assume the 3D shape belongs to a 'shape space', which is a linear combination of basis shapes [101]. Variants include trajectory prior basis other than shape [102], and closed-form solution of basis by adding constraints on basis [103]. Recently extensions include extending such method to multi-body cases [104].

Deformable models are widely used in 2D computer vision, including human face modeling [105], full-body modeling [106] and object detection [107], but it is not very frequently used in 3D. The reason is that one must make several assumptions on the deformation to solve these problems [103], which may limit the algorithm's application. We chose not to employ deformable SfM in crop monitoring application for two major reasons: (1) such method is not easy to extend to support multi-sensors input, and (2) they are all

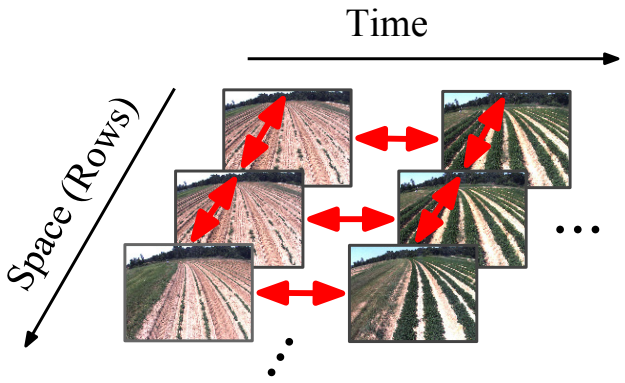


Figure 4.3: Data association (image matching) pattern of a 4D reconstruction. Each image represents a single row 3D reconstruction, and red arrows indicate performing image matching between single row datasets on each side.

derived from SVD factorization based method [50], so they are slow on large scene, and are limited to small scene applications like indoor human face or full-body tracking.

4.3 Data association over time and large baseline

Data association is a key technique to obtain 3D reconstruction results of more than a single row at a single time, however, the data association problems between different rows and times are difficult, since there are significant appearance changes due to illumination, weather or view point changes. As shown in Fig. 4.3, to build a multi-row 4D reconstruction, a.k.a align multiple single row 3D reconstructions at different rows and different sessions into a single coordinate frame, data association information is needed in *cross-time* and *cross-row* manners. We perform image matching to find visual correspondences only on nearby rows, and nearby sessions in data collection order (timer interval in data collection is smaller than a week).

Although both space/time intervals for cross-row/cross-time image matching are minimized by matching only nearby rows/sessions respectively (as shown in Fig. 4.3), there is still a significant amount of viewpoint variance that appears in cross-row image pairs, and appearance variance (caused by weather and illumination) that appears in cross-session im-

age pairs, makes the image matching problems difficult. The problems are even more challenging in crop monitoring due to *measurement aliasing* [108]: fields contain highly periodic structures with little visual difference between each plants (see raw images Fig. 2.3-2.5). As a result, data association problems between different rows and times is nearly impossible to solve by image-only approaches.

Rather than using an image-only approach, we employ a geometric information from the 3D reconstructions output by the SLAM pipeline in the proposed data association method as prior information. Since we can run all 3D reconstructions ahead of 4D reconstruction, the information available for cross-row and cross-time image matching is not only images, but also all 3D reconstruction results. The SLAM results provide camera poses and field structures from all of the sensors (not just images), which helps us to improve the robustness of data association.

4.3.1 Technical approach

Specifically, the data association problem involves finding visual correspondences (matches between SIFT feature points) between two images, I_1 and I_2, which are taken by camera C_1 and C_2 respectively, as shown in Fig. 4.4. Cameras C_1 and C_2 may come from the same or different rows, during the same or different time sessions. Each camera $C_i = \{\mathbf{K_i}, \mathbf{R_i}, \mathbf{t_i}\}$ contains the camera intrinsic calibration $\mathbf{K_i}$ which is calibrated offline, and camera transformation in global frame $\{\mathbf{R_i}, \mathbf{t_i}\}$ estimated by SLAM on a single row.

In this section we combine two methods to build a data association method using 3D prior information from the SLAM pipeline, *back projection bounded search* and *homography image warping*. Two methods are detailed below.

Back-projection bounded search The basic idea behind the back-projection bounded search is to reduce number of possible outliers by limiting the search range while seeking visual correspondences. Assume L_1 is the set of all estimated landmarks visible in C_1, and

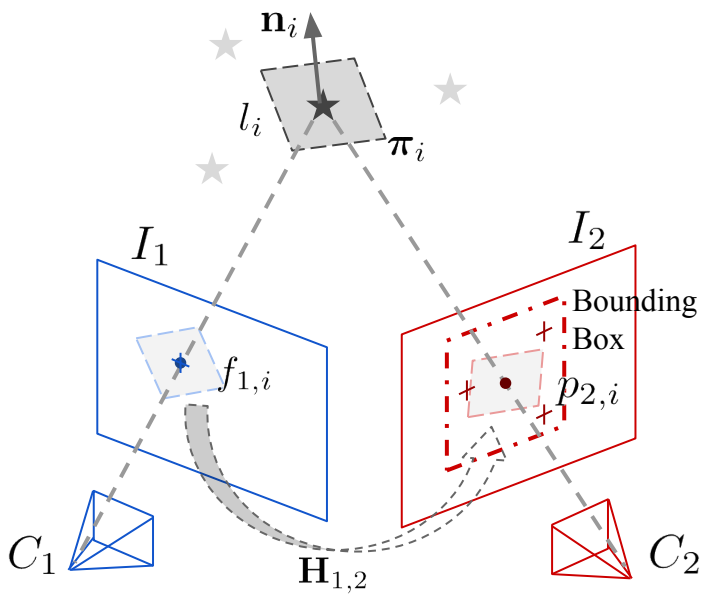


Figure 4.4: Robust data association diagram

each landmark in L_1 has corresponding feature points in I_1. For each $l_i \in L_1$, the linked feature point $f_{1,i} \in I_1$ might have a corresponding matched point at $p_{2,i} \in I_2$, which is the back-projected point of l_i on I_2, if C_1, C_2 and l_i are accurately estimated, and l_i has a corresponding visual feature keypoint extracted $p_{2,i}$ in I_2. But with estimation error exists in 3D reconstruction step, the back-projection is not accurate and the corresponding visual feature keypoint may offset from the beck-projected location.

To solve this issue, I define a relaxed search area by a bounding box centered at $p_{2,i}$ on I_2, as shown in Fig. 4.4, to search for the corresponding feature point for $f_{1,i}$. This significantly limits the search area to match $f_{1,i}$ (compared with searching the whole I_2 to a limited bounding box area, whose size is about 50×50 pixels), and rejects many possible outliers, meanwhile still maintains most inlier matches which have reprojection error smaller than the bounding box size, makes the later outlier rejection step easier.

Homography image warping Although the back-projection bounded search rejects the majority of possible outliers, the data association problem is still difficult when the viewing angle changes: the object's appearance may change significantly with large camera

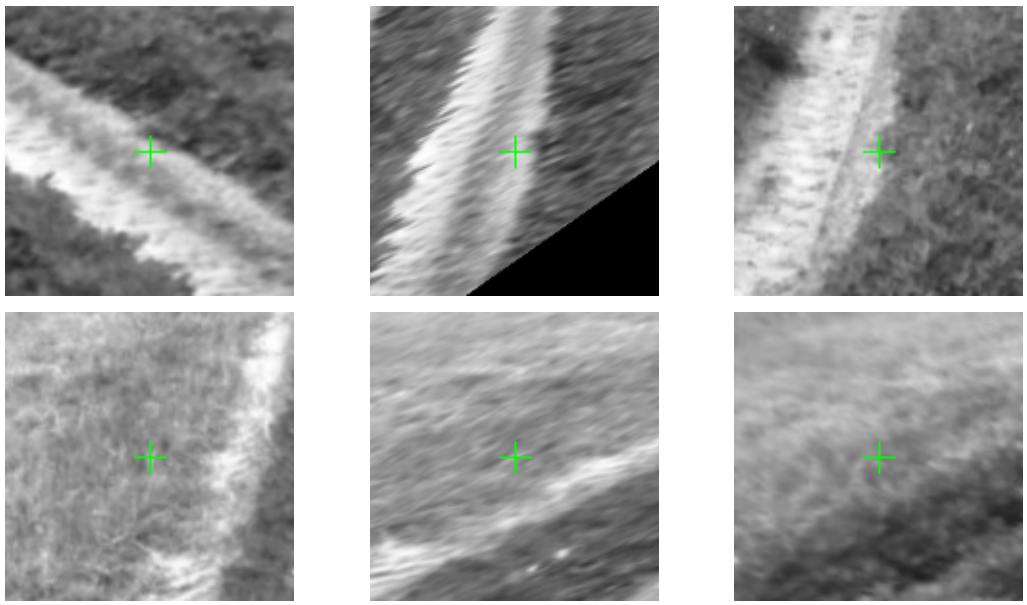
baselines, which changes the visual descriptors of the same object and makes the matching difficult. This is the major challenge for data association across images collected in different rows.

To solve this issue, a homography-based method is used to eliminate appearance variations in l_i due to viewpoint changes. This is partially inspired by [109], but with the major difference that our proposed method uses feature points instead of patches. We assume that l_i lies on a local *plane* $\boldsymbol{\pi}_i$. If this assumption is satisfied, $\boldsymbol{\pi}_i$ induces a homography $\mathbf{H}_{1,2}$ from I_1 to I_2 [86, p.327]

$$\mathbf{H}_{1,2} = \mathbf{K}_2\big(\mathbf{R}_{1,2} - \frac{\mathbf{t}_{1,2}\mathbf{n}_i^\top}{d}\big)\mathbf{K}_1^{-1} \tag{4.1}$$

where $\{\mathbf{R}_{1,2}, \mathbf{t}_{1,2}\}$ define the relative pose from C_1 to C_2, $\mathbf{n}_i$ is the normal vector of $\boldsymbol{\pi}_i$, and d is the distance from C_1 to $\boldsymbol{\pi}_i$. We use $\mathbf{H}_{1,2}$ warp I_1 to get I_1', which has same view point with I_2, and thus similar appearance. Next a SIFT descriptor $f_{1,i}'$ on wrapped I_1' is extracted for bounded search rather than using the original $f_{1,i}$. Two example patches are shown in Fig. 4.5: although I_1 and I_2 have significant appearance variation, since they are taken from different rows, the warped I_1' has a very similar appearance to I_2, which makes SIFT descriptor matching possible.

The whole data association process is described below. First normal vectors $\mathbf{n}_i$ of all points in local landmark point cloud around l_i in L_1 are estimated. Homography image warping is only enabled when the baseline between C_1 and C_2 is longer than a threshold, in our system this is set to 0.5m. After getting all nearest neighbor feature matches by back projection bounded search, a final outlier rejection is performed by 8-point RANSAC. The full data association pipeline is summarized in Algo. 1.



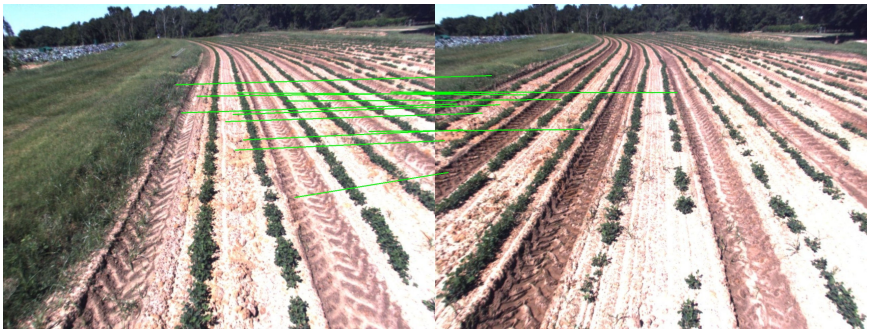
(a) I_1 (b) I_1' (c) I_2

Figure 4.5: Homography image warping on two random images patches. (a) original I_1, (b) warped image I_1', and (c) original I_2. Patch center with green cross is feature point $f_{1,i}$ on (a), $f_{1,i}'$ on (b), and back project point $p_{2,i}$ on (c).

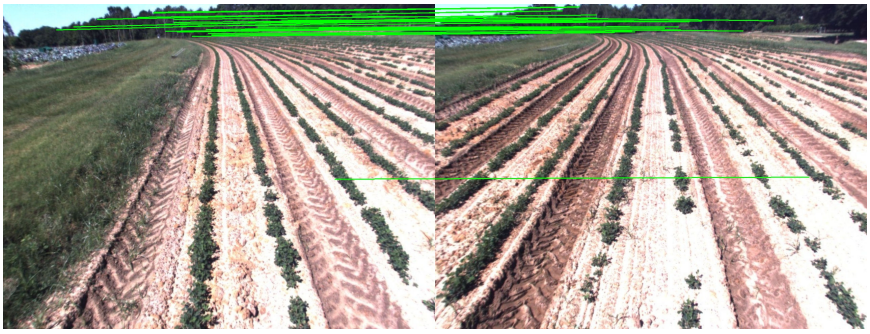
4.3.2 Evaluation

The proposed data association approach is evaluated using realistic data provided by our field dataset. As discussed in 4D reconstruction problems, there are two types of data association problems are too hard for appearance-only method: (1) when the view point changes a lot, which happens when the camera looks at the same object at different places of field, and (2) when the images are taken at different times (weeks). To show the effectiveness of proposed method, we setup a appearance-only baseline algorithm to compare with: SIFT features are extracted from two images first, then the SIFT features are matched by approximate nearest neighbor method FLANN [85], and filter outlier by 8-point RANSAC.

The experimental results validate the performance and robustness of the proposed approach. Cross-row (1st vs. 3rd row) data association results are shown in Fig. 4.6. The naive FLANN+RANSAC approach can only recover feature matches on the top, where far away objects do not change their appearance, and it fails to register any crops correctly

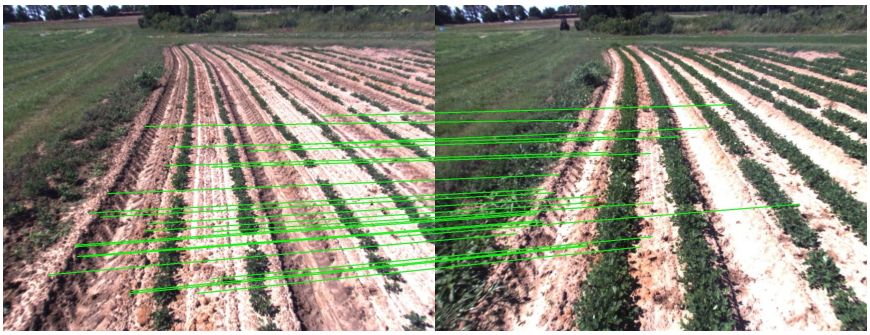


(a) Proposed method



(b) SIFT+FLANN+RANSAC

Figure 4.6: Cross-row image matching example results of a image pair between 1st and 3rd row of June 9, 2016. Only inlier matches are shown in green.



(a) Proposed method

(b) SIFT+FLANN+RANSAC

Figure 4.7: Cross-time image matching example results of a image pair between 1st row of June 9 and June 20, 2016. Only inlier matches are shown in green.

Algorithm 1: Robust data association method

Input : Image I_1, I_2, Camera C_1, C_2, Landmarks L_1
Output: Set of matched feature point pairs $P_{1,2}$
set match set $P_{1,2} = \emptyset$
foreach $l_i \in L_1$ **do**
  back project l_i to $C_2 \rightarrow p_{2,i}$
  if C_1 *and* C_2 *baseline length* $<$ *threshold* **then**
    $f'_{1,i} = f_{1,i}$
  else
    calculate homography $\mathbf{H}_{1,2}$ use Eq. 4.1
    use $\mathbf{H}_{1,2}$ warp $I_1 \rightarrow I'_1$
    calculate SIFT descriptor at $p_{2,i}$ on $I'_1 \rightarrow f'_{1,i}$
  end
  set l_i's match set $P_i = \emptyset$
  foreach *feature point* $f_{2,j} \in I_2$ **do**
    if $f_{2,j}$ *in bounding box of* $p_{2,i}$ **then**
      insert $[f'_{1,i}, f_{2,j}] \rightarrow P_i$
    end
  end
  find min $L2$ of SIFT descriptor in $P_i \rightarrow [f'_{1,i}, f_{2,k}]$
  insert $[f_{1,i}, f_{2,k}]$ into $P_{1,2}$
end
RANSAC_8pt_reject_outlier($P_{1,2}$)

in the field. However, the proposed approach can register feature points in the field, with significant changes of appearance. Similar results are shown for cross session matching in Fig. 4.7, that more inlier matches are recovered in the field.

4.4 Spatio-temporal (4D) reconstruction

The goal of the 4D reconstruction is to perform registration between multiple 3D point clouds that are obtained at different times of the growing season, and align them into a single coordinate frame. Registrations between 3D point clouds are traditionally solved by estimating rigid body transformation between point clouds by ICP [110] or recently by mutual information [111], but when you are looking for high precision, estimating a single rigid body transformation is not a good idea: minor error in rotation estimation will

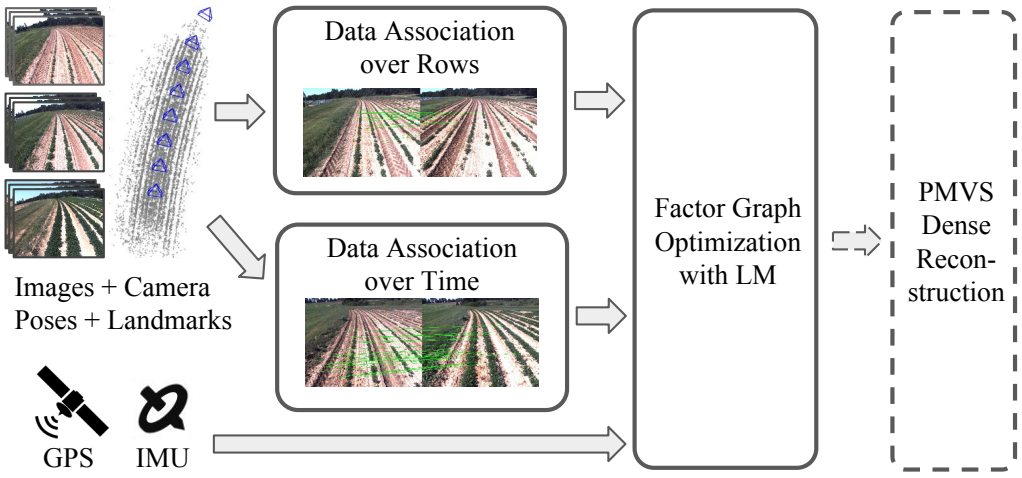


Figure 4.8: Overview of multi-sensor 4D reconstruction pipeline. Dash box of PMVS part indicates it is optional.

cause significant translational error far away of rotation center, and different point clouds may suffer non-rigid transformation errors, which cannot be fixed by a single rigid-body transformation. In this section we propose a factor-graph based method which can solve the problems of rigid-body transformation.

4.4.1 Technical approach

Fig. 4.8 shows the diagram of the proposed 4D reconstruction pipeline. The input of the pipeline is all the raw sensor data, plus multiple 3D reconstructions (estimated camera poses and landmarks) from multi sensor SLAM. The first step is to build cross-row and cross-time image matching, follows the pattern defined in Fig. 4.3. After build data associations, the 4D reconstruction is performed by optimizing a 4D factor graph. Finally optional dense reconstructions could be obtained from PMVS [112].

Before formally introducing the definition of the 4D factor graph, we explain how cross-row and cross-time visual correspondences can be used to build ”shared landmarks”, after which we can construct the build 4D factor graphs. As shown in Fig. 4.9, given two 3D reconstructions (shown in red and blue) which are captured at different time or different rows, we use the proposed robust data association pipeline to match image features. For

a single inlier feature matches (a green line shown), if both 2D feature points have their corresponding 3D landmarks in 3D reconstructions respectively (shown by dash arrows), these two 3D landmarks actually correspond to the same 3D location, because they are matched by appearance.

After determining all the 3D landmark pairs that correspond to the same 3D locations, *shared landmarks* are obtained to connect factor graphs for 3D reconstructions, to build a *4D factor graph*, as shown in Fig. 4.10. Given a 3D landmark pair which correspond to same 3D location, we can merge two landmarks into a single landmark, called "shared landmark", which appear in both 3D reconstruction. Shared landmarks play roles like anchors, which connect multiple 3D reconstructions into a single frame. The 4D factor graph is built by multiple multi-sensor SLAM factor graphs, but connected by shared landmarks, as shown in Fig. 4.10.

The formal definition of a 4D optimization is the joint estimation process of all camera states $X = \bigcup_{t_i \in T, r_j \in R} X^{\langle t_i, r_j \rangle}$ and all landmarks $L = \bigcup_{t_i \in T, r_j \in R} L^{\langle t_i, r_j \rangle}$, where R and T are set of rows and sessions, respectively. The measurements $Z = \bigcup_{t_i \in T, r_j \in R} Z^{\langle t_i, r_j \rangle} \cup Z_{cr}$ includes all single row information as well as data association measurements Z_{cr} that connect rows across space and time. From here if not mentioned separately, like in last chapter, we still use i to index different dataset sessions, j to index different rows in a single session, k and l to index different camera states and sensor measurements in a single session an a single row respectively.

Similar to the proposed multi-sensor SLAM from the previous chapter, the joint probability of all camera states are defined by

$$p(X|Z) \propto \prod_{t_i \in T} \prod_{r_j \in R} \phi(X^{\langle t_i, r_j \rangle}) \prod_{h=1}^{H} \phi(X_{cr,h}), \tag{4.2}$$

where H here is the size of Z_{cr}, and $X_{cr,h}$ is the set of states hth measurement of Z_{cr} involved. This joint probability can be expressed as a factor graph, shown in Fig. 4.10

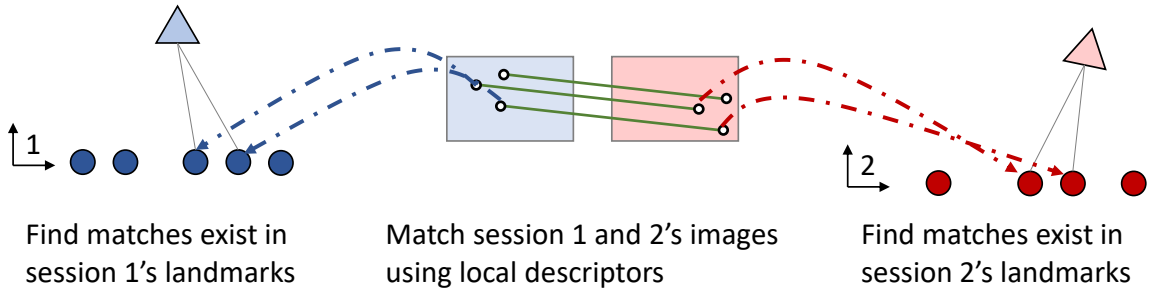
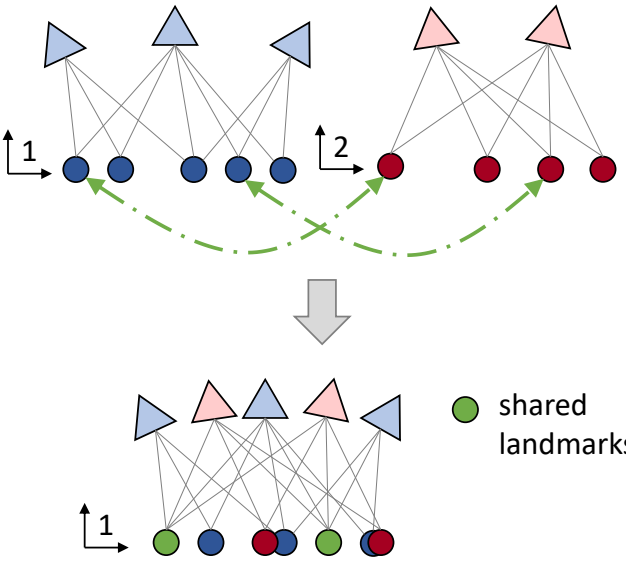
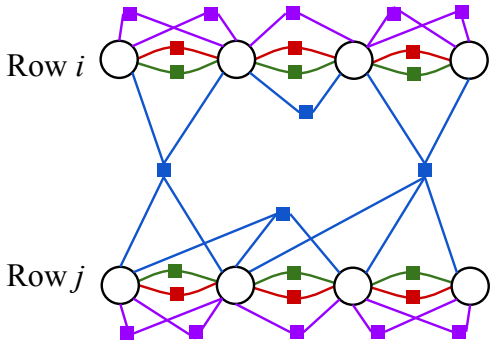


Figure 4.9: Illustrating how to use cross-time (or cross-row) image matching to build data association between two 3D reconstrucions. If each 2D point in a matched feature pair (shown in green) has corresponding 3D landmarks in two 3D reconstruction, these two landmarks matches each other.



(a) Shared landmarks from cross-time image matching.



(b) Factor graph of multi-sensor 4D reconstruction.

Figure 4.10: Illustrating how to use shared landmarks to build 4D reconstruction. (a) Get shared landmarks from cross-time image matching, matched image features are marked in green and green landmarks are shared landmarks. (b) Example factor graph of 4D reconstruction, shared landmarks are blue factors cross sesssions.

(b). The first part of the joint probability consists of the factor graphs from all single rows. And the second part is the cross-row and cross-session measurements Z_{cr}, which are vision factors generated from cross-row and cross-session data association. We call the added factors *shared landmarks*, since they are shared by two (or possibly more) rows, and they have two (or more) sessions associated with them. Data association is performed across different rows and times to get Z_{cr}. Exhaustive search between all row pairs is not necessary, since distances or timespans that are too large make data association impossible. In our approach, we only match rows next to each other in either the space domain (near-by rows in the field), or the time domain (near-by date) using the proposed robust data association approach, as shown in Fig. 4.3.

Solving the MAP estimation problem gives the estimated camera states

$$\hat{X} = \underset{X}{\operatorname{argmax}}\, p(X|Z). \tag{4.3}$$

the Levenberg-Marquardt algorithm is used to solve the optimization problem, with initialization from the results of multi-sensor SLAM. Outlier rejection of vision factors [88] is also enabled during optimization, to reject possible false positive feature matches from cross-row and cross-session data association. Landmarks $\hat{L}$ are estimated by triangulation given estimated camera poses.

The point cloud $\hat{L}$ obtained by the MAP estimation is relatively sparse, since it comes from a feature-base SLAM pipeline, in which only points with distinct appearance are accepted as landmarks (in our system SIFT key points are accepted). An optional solution to solve this issue is to use PMVS [112], which takes estimated camera states $\hat{X}$ to reconstruct dense point clouds.

4.4.2 Evaluation

The proposed approach is implemented by the GTSAM C++ library.[1] RTK-GPS data is used from the peanut dataset as GPS input, and ignored lower accuracy GPS data. Since the tractor runs back and forth in the field, only rows in which the tractor driving south (odd rows) are used, to avoid misalignment with reconstruction results from even rows.

Fig. 4.11 shows example densely reconstructed 4D results output by PMVS. Although Fig. 4.11 shows that the 3D reconstruction results for each single session qualitatively appear accurate, to make these results useful to precision agriculture applications, are interested in evaluating the approach quantitatively. In particular we wanted to answer the following questions:

- Are these 3D results correctly aligned into a single frame in space?
- Are these 3D results useful for measuring geometric properties of plants useful for crop monitoring (height, width, etc.) ?

To answer the first question, we show an enlarged part of an example 4D sparse reconstruction in Fig. 4.12. The figure shows that all 3D point clouds together in a *single* coordinate frame, which the 4D model is estimated (since we are using GPS data, it is the world frame). Point clouds from different dates are marked in different colors. We can see from the cross section that the ground surface point clouds from different sessions are aligned well, which shows that all of the 3D point clouds from different dates are registered accurately into a single coordinate frame. This suggests that we are building a true 4D result. We can see the growth of the peanut plants, as the point cloud shows 'Matryoshka doll' like structure, earlier crop point clouds are inside of point clouds of later sessions.

To compare our approach with the existing ICP method, Fig. 4.12 shows the 3D point clouds aligned by ICP [110] for comparison. The ICP results are significantly worse than alignments computed by our proposed approach, since ICP can only compute a single rigid

[1] `https://bitbucket.org/gtborg/gtsam`

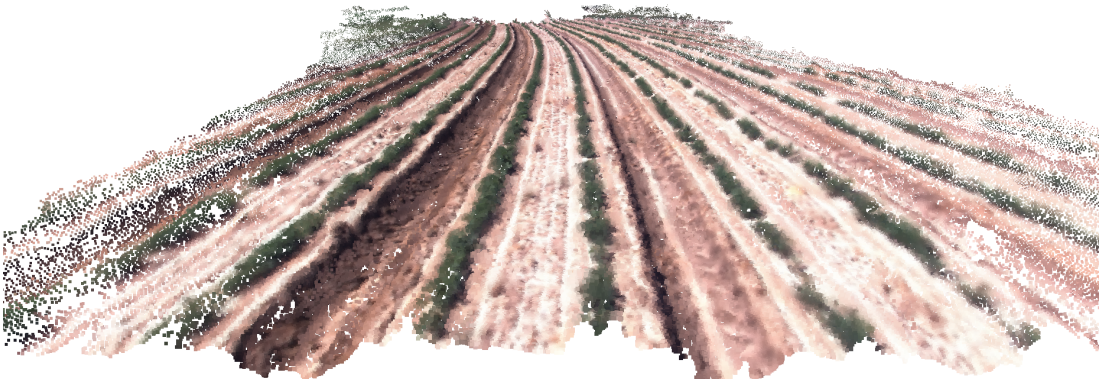
relative transformation for each point cloud pair. In contrast, the proposed approach can perform data association in multiple places in the field, which is equivalent to a 'non-rigid' transformation.

To answer the second question, we calculate the example crop height results using reconstructed 4D point clouds, and compare the results with the ground truth in the form of manual height measurements. A simple pipeline is set up to estimate the height of peanut plants from sparsely reconstructed 4D point clouds at multiple sites, by (1) estimating local ground planes by RANSAC from May 25's point cloud for each site (when peanuts are small and ground plane is well reconstructed); (2) separating the peanut canopy's point cloud by color (using RGB values); and finally (3) estimating the distance from the peanut canopy's top to the local ground plane.

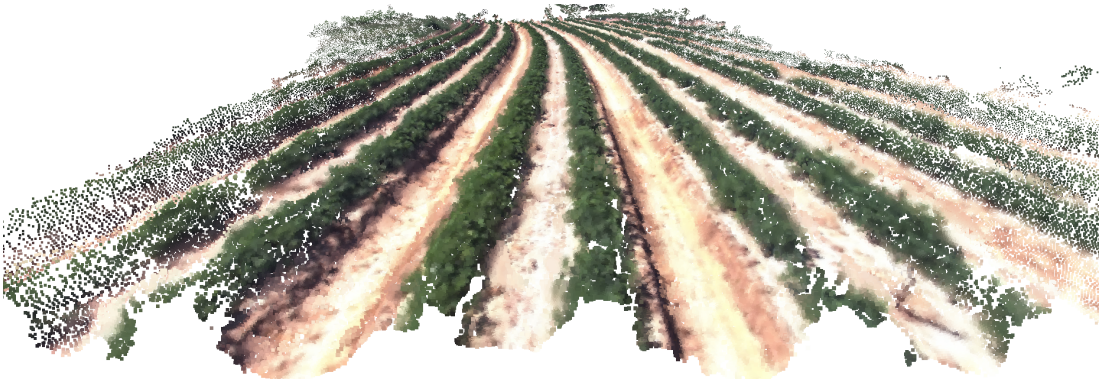
Fig. 4.13 shows height estimations during the 2016 growing season at 12 sampling sites. Except for exceptions of sites 22 and 25, which have slightly biased height estimates due to poor RANSAC ground plane estimates, the results match the ground truth measurements well. For all of the sites, the root-mean-square(RMS) error of height estimation is 2.93cm. This is better compared to reported performances of LIDAR based methods [8], shows that we can compute reasonable height estimates even with a simple method, and proves that the 4D reconstruction results contain correct geometric statistics.

4.4.3 Application in precision agriculture

4D point clouds of crops provide qualitative geometric information of crops, and they are useful for visualization and high-level decision making purpose, but they are not directly useful in precision agriculture applications that need accurate quantitative geometric information of crops. To help precision agriculture applications, 4D models need to be further processed to get information useful for farmers, like crop heights, canopy sizes, etc. In this section two types of example post processing techniques based on 4D reconstruction are discussed, which are useful for farmers: crop height histogram time series of the fields, and



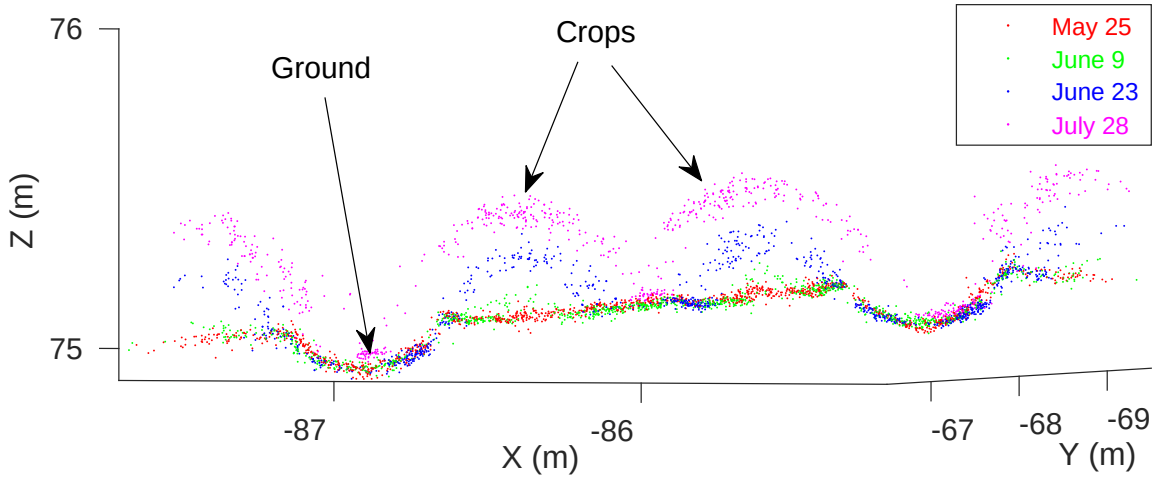
(a) June 9



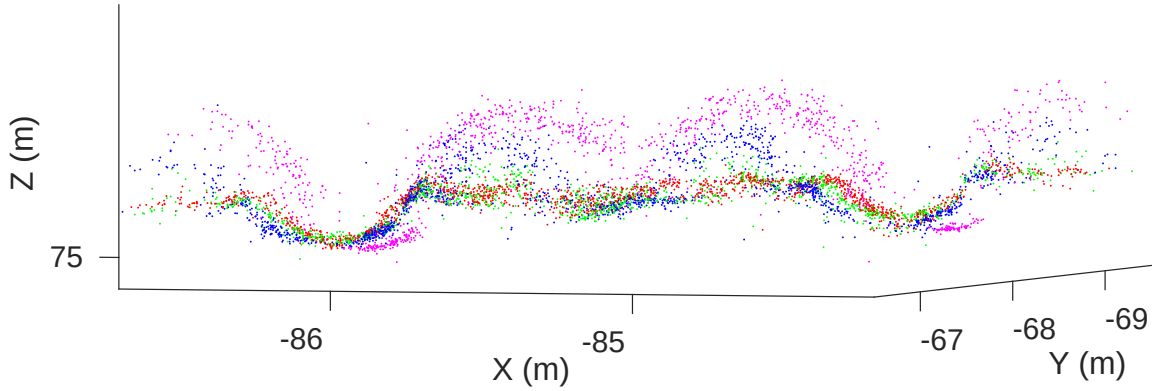
(b) June 23

(c) August 1

Figure 4.11: Example 4D PMVS reconstruction results. Three dense point clouds are output by PMVS at three dates in 2016.



(a) Proposed method



(b) ICP [110]

Figure 4.12: Cross section of part of the sparse 4D reconstruction results at 3rd row. Upper subfigure is results of proposed approach, lower subfigure is results of ICP [110]. Ground surface is marked in upper subfigure. Only 4 sessions are shown to keep figure clear.

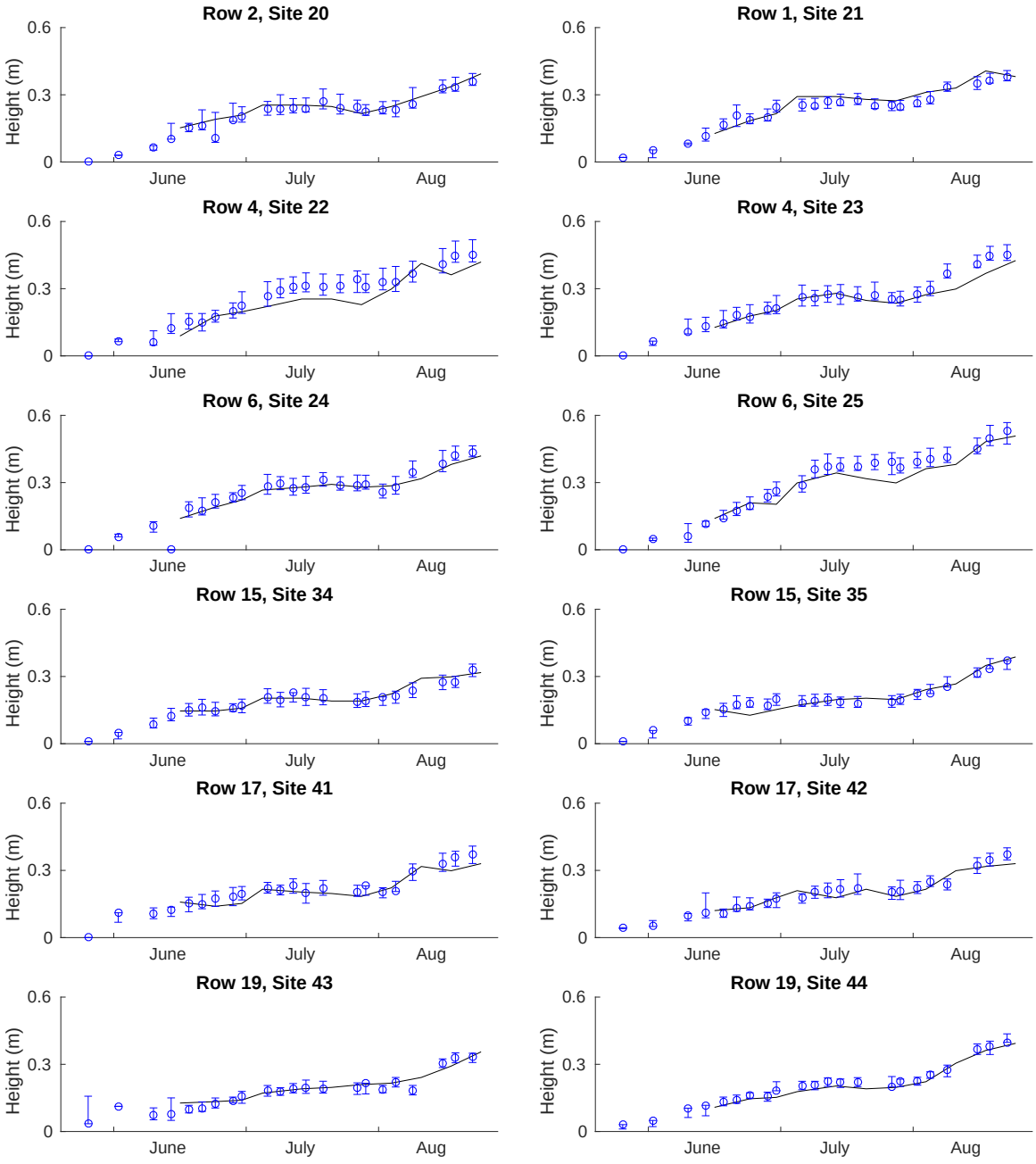


Figure 4.13: Estimated peanut heights at 12 sampling sites in blue, with ground truth manual measurements in black lines.

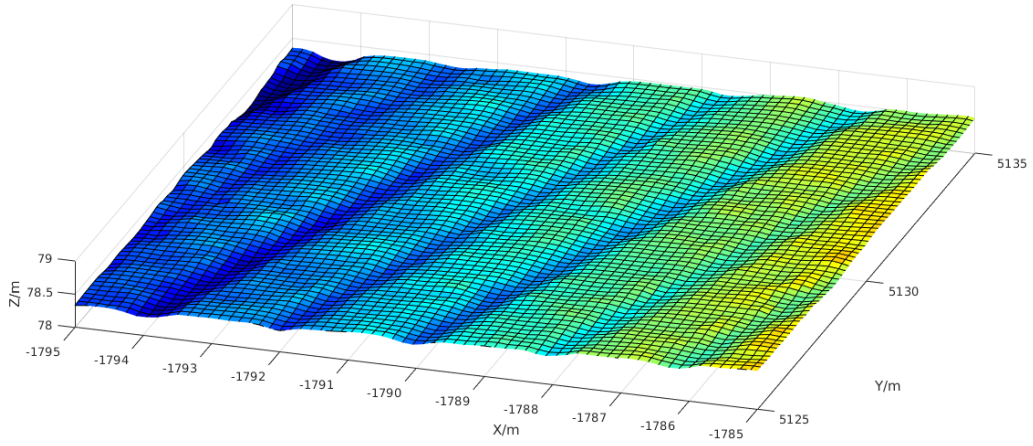


Figure 4.14: Example fitted ground mesh, the absolute height is makred by color coding.

canopy sizes estimation by an expectation-maximization based method.

Crop height histogram time series One of the most useful statistics farmers can obtain from the field 3D information is the height of crops. In the previous section we have already shown some crop height estimation results at specified locations in the field in Fig. 4.13. But the method to get these results is limited to be applied at 1D point locations, not easily to be applied at whole fields.

To solve the issue above, I propose a histogram-based method to estimate crop height of the whole field over a field 3D point cloud, and the method can be easily extended to calculate time series of height histogram from a 4D point cloud. For a 3D point cloud, a single crop height histogram is calculated by the following protocol: (1) ground mesh estimated by grid regularization on the 3D point cloud of first session (which collected before seeding and has only bare soil with no crop), then (2) calculate the z-axis distances to ground mesh of all points in the point cloud, and finally (3) divide the field into a grid of 0.2m$\times$0.2m resolution at x-y plane, and calculate the height of each cell by averaging all the z-axis distances of points in this x-y cell.

Since we want to calculate the heights of the crops relative to the ground plane, we need to remove the height information corresponding to the terrain from the point cloud.

If we assume the ground plane does not change during the whole growing season (which is a reasonable assumption, since changes of the terrain caused by rain wash and human activities are negligible), the ground mesh before growing season can be used as current ground mesh, and used to calculate crop heights. Fig. 4.14 shows an example fitted ground mesh, with absolute height color coding. The row structure of the field is clearly shown, and it's clearly shown that the right side of the field is higher than the left side, means the field the tilted.

Fig. 4.16, 4.17 and 4.18 show the resulting time series of height histogram for the years 2016, 2017 and 2018 respectively. In the 2016 results, the uneven pattern of heights is caused by magnesium deficiency. In the 2017 results, the uneven pattern is mainly caused by compacted soil last year, that the curb and road areas in 2016 have lower crop height in 2017.

Canopy bounding box estimation In precision agriculture applications, farmers may not only care about crop height, but also about other 3D geometry information, like volume, canopy width, etc., and these quantities cannot be provided by the height maps. To provide crop geometry information more than just height, Carlone et al. [113] proposed an expectation maximization (EM) based method, which takes proposed 4D model as input, estimate time series of bounding boxes of crops. In the reconstructed point clouds as shown in Fig. 4.11, different plants have significant overlaps with each other, so the proposed EM based method assigns a latent variable for each point in the point cloud, and estimates which plant the point belongs to. Fig. 4.15 shows example estimated bounding boxes and canopy sizes estimated from proposed dense 4D point clouds.

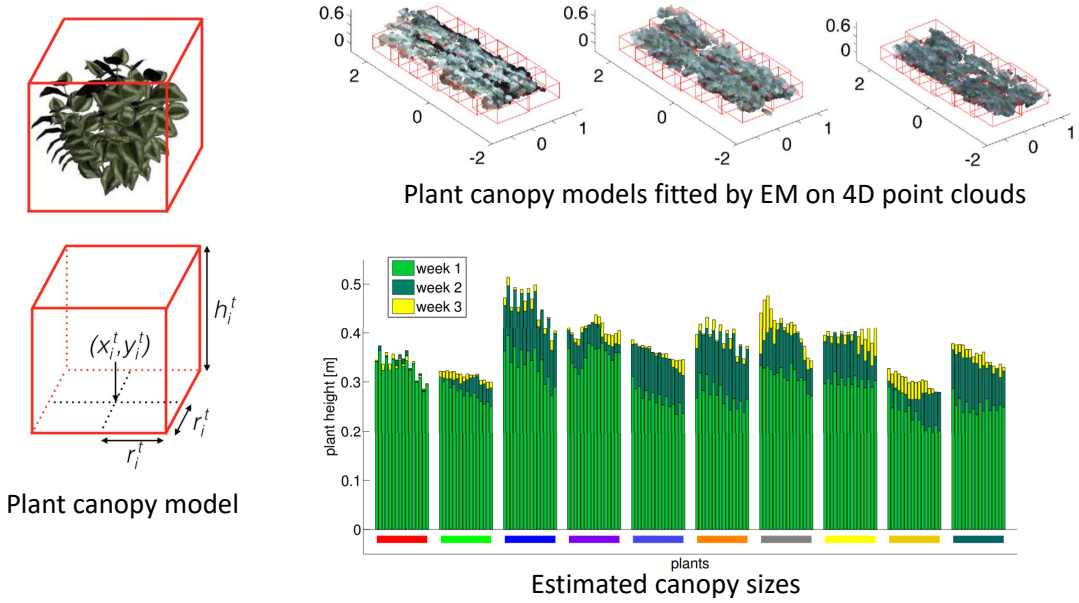
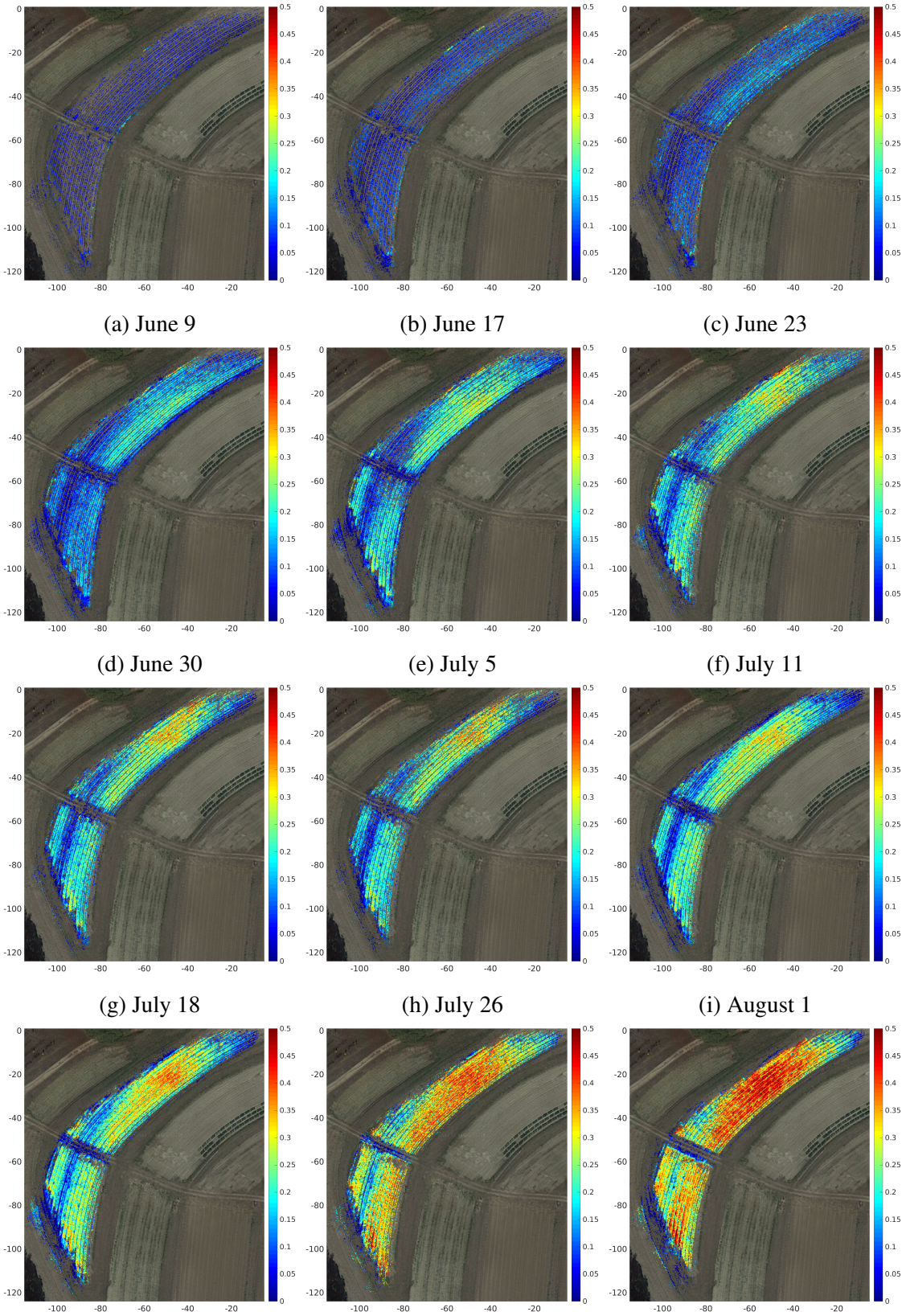


Figure 4.15: Results of bounding boxes estimation. Left is the bounding box parameters estimated by EM, upper right shows estimated bounding boxes, and lower right shows estimated canopy size series of three example weeks.



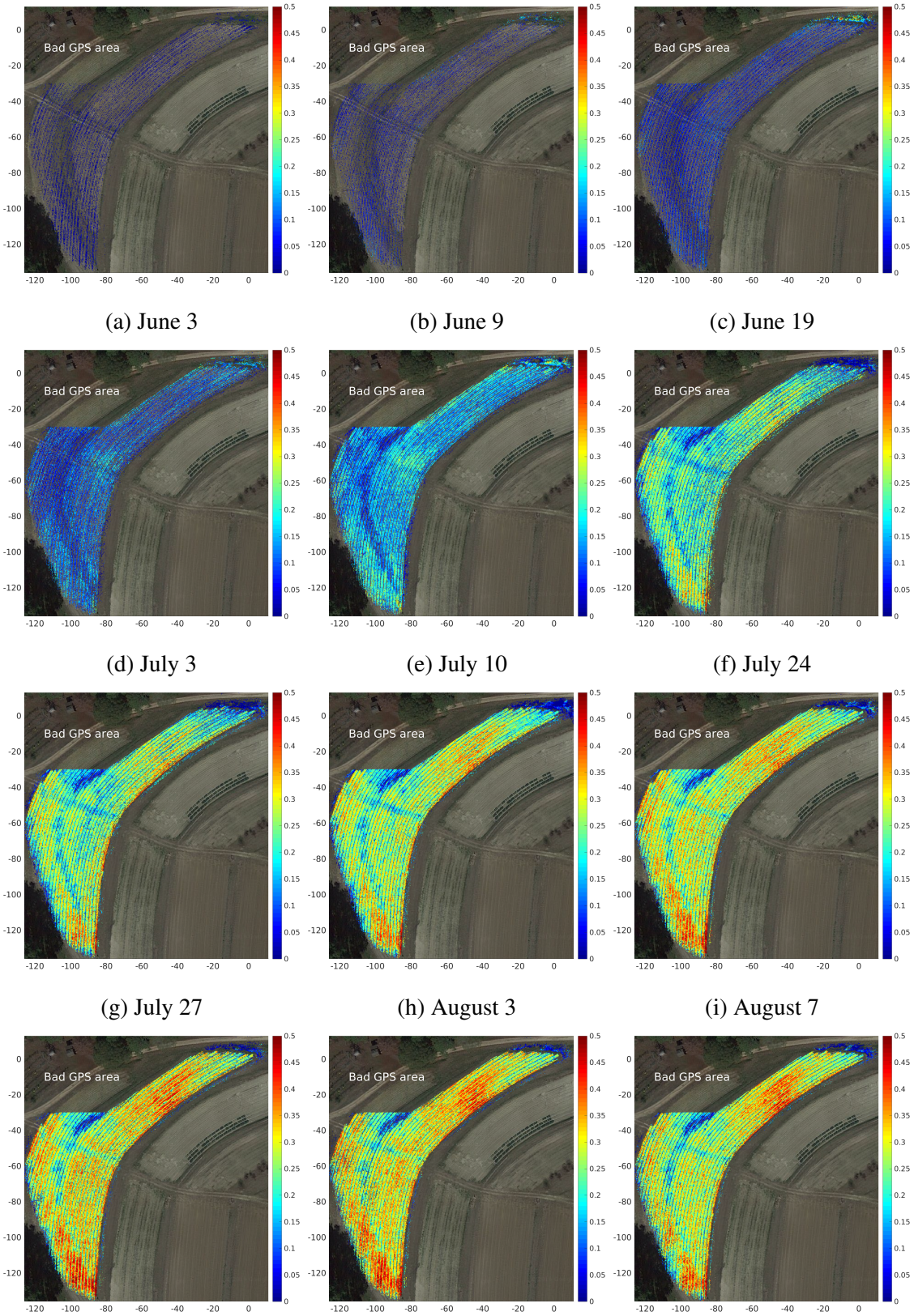
(a) June 9 (b) June 17 (c) June 23

(d) June 30 (e) July 5 (f) July 11

(g) July 18 (h) July 26 (i) August 1

(j) August 8 (k) August 15 (l) August 22

Figure 4.16: Time series height histogram of year 2016.



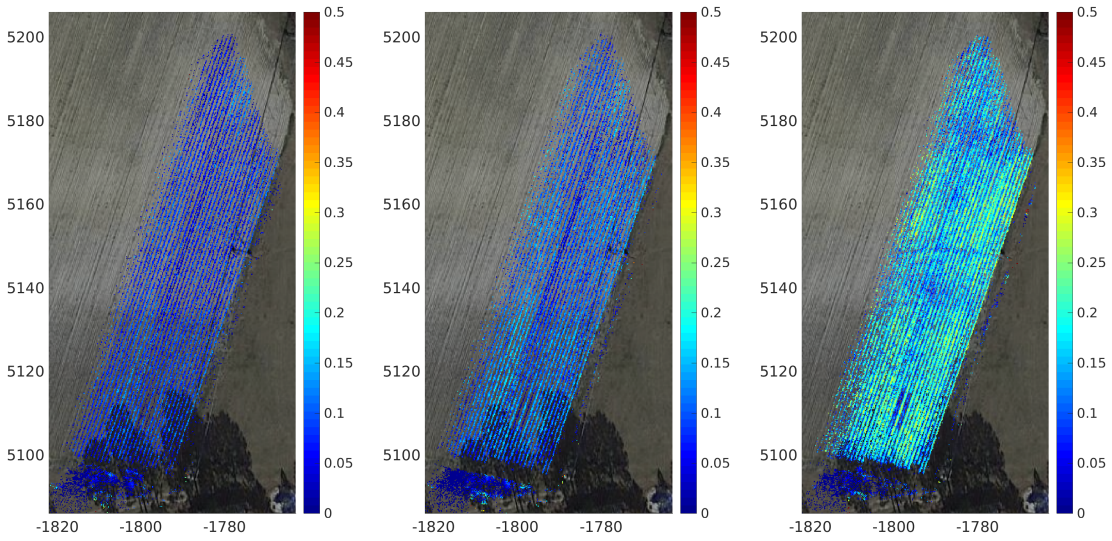
(a) June 3 (b) June 9 (c) June 19

(d) July 3 (e) July 10 (f) July 24

(g) July 27 (h) August 3 (i) August 7

(j) August 14 (k) August 21 (l) August 31

Figure 4.17: Time series height histogram of year 2017.



(a) July 3 (b) July 9 (c) July 23

Figure 4.18: Time series height histogram of year 2018.

CHAPTER 5
WEAKLY-SUPERVISED CROSS-MODALITY IMAGE DESCRIPTOR LEARNING

5.1 Introduction and claims

In the previous chapters of this thesis, techniques to build 3D and 4D reconstructions were proposed, which can be applied to RGB or images of any modality, but we do not have the abilities to obtain correspondences between 3D/4D reconstructions across different modalities, which can also provide useful information in precision agriculture. For example, if we care about the color and near-infrared reflection information of a particular 3D spot, we need the RGB 3D/4D reconstruction and the near-infrared 3D/4D reconstruction *at the same location*. Similar to the 4D reconstruction problems, getting 3D/4D information of different image modalities needs to ability to associate 3D/4D reconstructions from different modalities into *a single coordinate frame*. To register 3D reconstructions from multiple modalities, we need to be matched to build visual correspondence between 2D images which are in multiple different modalities.

Registering images across different modalities is not a task that is limited to 3D reconstruction. The calculation of Normalized Difference Vegetation Index (NDVI) [35] which is used to evaluate vegetation in remote sensing images needs near-infrared and red illumination densities of each pixel, which needs to register and align red and near-infrared images.

Although registering and matching images across different modalities are fundamental tasks in images processing of multiple modalities, researchers have not found a general solution yet. Some multi-spectral cameras have calibrated aligned lens array for different frequencies to make the output images aligned automatically, but those cameras with lens

array are generally more expensive and need accurate manual optical calibrations.

The standard approach to register images is by matching local descriptors in two images using some type of metrics. People started doing this with hand-crafted local image descriptors in early 2000, and SIFT [84] is the baseline method for image registration and matching. But most of the hand-crafted local image descriptors do not work out-of-box on multi-spectral and hyper-spectral agriculture image sets, since in different image modalities gradient direction may reverse: area brighter in RGB images may darker in infrared images, or vice versa. People have spent lots of efforts on hand-crafted local image descriptors, make modifications to make them work on multi-spectral and hyper-spectral cases, but most of them are application-targeted approaches and not general enough for other applications.

Learning-based descriptors, especially convolutional neural network (CNN) based descriptors, have drawn a lot of attention in recent years because of their superior performances without the need for sophisticated hand-crafted feature design skills. Another reason why learning based methods are preferred in cross-modality registration and matching is that given enough training data, these methods are adaptive between any modalities. But one of the major drawbacks of most learning based local image descriptors is that they need lots of training data from manual labeling, and the trained models do not apply to other image modalities. This limits the usage of learning based methods, since it's user's responsibility to collect a large amount of data for training, and the data collection may be hard due to required large amount of data.

In this chapter I propose a weakly-supervised method to train CNN local image descriptors for matching pairs of images across different modalities. The advantage of the proposed method is that the transformation between the input image pair is automatically estimated during the training process of the CNN, with only a single inaccurate initial image alignment parameter, and no precise manual transformation or patch pair supervision input is required. The proposed method saves lots of labor and cost to train descriptors

for cross-modality matching purpose, makes it useful in real applications, since no precise image alignment or large amount of image patch pairs is required, using

Although the cross-modality image registration and matching method is proposed for multi-spectral image registration purpose, the *modality* here is not limited to image spectrum (frequency): it can be any inherent property of the image, e.g. illumination, viewpoint, etc. So we can also apply the proposed method to image registration and matching problems other than multi-spectral registration problems: data association between different image modalities: different illumination caused by different weather, shadows, time of the day, and viewpoints, etc.

My claims for the proposed method are:

- *Comparable-to-supervised*: the proposed method only need inaccurate human supervision on transformation between image pairs, and get comparable performance with supervised methods, save lots of labor from accurate annotation.
- *General*: the proposed method is general to image registration and matching problems between any image modalities, for example, different electromagnetic frequency, different illumination caused by weather/shadows/time, different viewpoints, etc.
- *Robust*: the proposed method is robust to large appearance difference between image modalities, and large noise in input supervision.

This chapter summarizes and extends materials from my previous publication [114].

5.2 Related work

Hand-crafted local image descriptors Hand-crafted local image descriptors have been invented and used for more than a decade. Most widely used hand-crafted local image descriptor include SIFT [84] and SURF [115]. They are all using image local gradient information to calculate the descriptors, while SURF gains speed up than SIFT by approximating Gaussian second order partial derivatives by box filters. One of the problems of

SIFT and SURF descriptors is that comparing them needs to calculate L2 distance, which is slow to compute. So a new type of descriptor called binary descriptor is invented, which addresses the issue by computing Hamming distance (can be efficiently computed on hardware) during comparison. Example binary descriptors include BRIEF [116], BRISK [117] and ORB [118]. Hand-crafted descriptors for cross-modalities image registration have also been studies including shape context [119] and self-similarities descriptor [120].

Learning based local descriptors Siamese networks [121] are widely used to train image local descriptors. A siamese network is a type of network applying symmetric (shared-weight) CNN on both images of input image patch pair, and it has been used to train CNN based local image descriptors. People have spent effort on formulating better loss functions for siamese networks to get improved performance, for example applying Hinge embedding loss [122], or by hard mining [123]. In addition, the papers [124, 125, 126] have proposed a variant on the siamese network, called triple network, and make images triplets as network input to improve the performance. Applications of siamese networks and their trained image local descriptors include image patch matching [127], stereo matching [128, 129], optical flow [122] and robot localization [130].

Siamese networks have been also used in direct end-to-end image matching [131, 127, 132] other than local iamge descriptor. We do not favor such an approach, since we need the raw image local descriptor to (1) approximate nearest neighbors (ANN) [133] like Kd-tree or FLANN [133], which speed up the brutal force image matching process, and (2) utilize other fast-search data structures, including Bag of word and vocabulary tree [134].

Cross-modalities image registration Mutual information has been used for cross-modalities image registration in medical imaging for years [135] including registration across CT and MRI, and there are existing software package [41]. A complete survey of using mutual information in medical imaging is [135]. Remote sensing community has also studied cross-modalities image registration for satellite multi-spectral images. Existing papers in-

clude hand-crafted modification SIFT for multi-spectral image matching [39, 42], spatial information helps matching original SIFT [40], and combine SIFT with mutual information [136]. Other than image registration, mutual information has been also used for point cloud registration [111].

Cross-modalities sensor data registration by neural network is proposed in [137], which uses a neural network similar to siamese network. This is similar to the proposed method, but it is limited to laser data and image registration, not cross-image registration. Also this method is not an unsupervised or self-supervised approach.

5.3 Weakly-supervised cross-modality descriptor learning under homography

The purpose of weakly-supervised descriptor learning is to learn two image local descriptor functions for each image, given a pair of image which may or may not have different image modalities. The term *weakly-supervised* comes from the nature of the problem that the only input of the proposed descriptor learning approach is a pair of image which may or may not have same image modalities, and does not need to accurately aligned or registered, plus a *rough* alignment. The output is an accurately estimated image alignment parameter, plus a CNN descriptor network which can be used to generate $L2$-distance matchable descriptors for the image pair. The idea of weakly supervised descriptor learning is shown in Fig. 5.1.

The image modality may have different definitions in different application. Example image modalities include spectral frequencies or sensor types (RGB, near-infrared , far-infrared, etc.), illumination conditions (day or night, different shadows, etc.), or other possible reasons cause the same object gives variant appearances in different images.

The use of the algorithm discussed in this section is only limited to the cases that exists *a single parameter* defines warping between the two images of image pair. Formally says, given input image pair I and I', here exists a warping function $\mathbf{w}(\cdot;\psi)$ with parameter ψ,

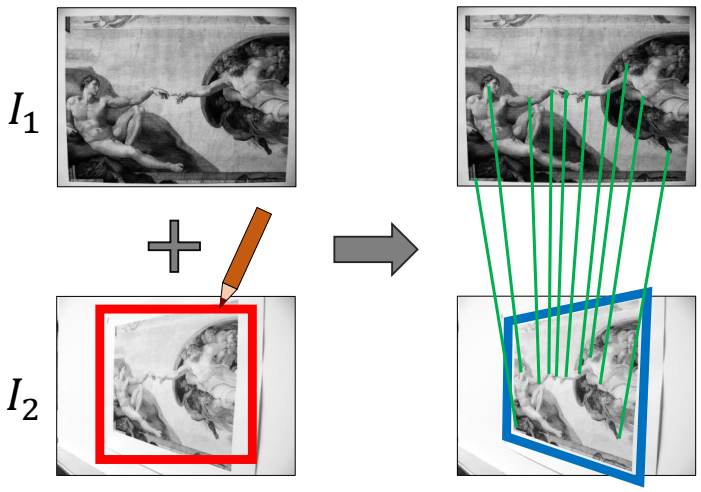


Figure 5.1: Idea of Weakly-supervised descriptor learning. Left is input and right is output. Red box is the rough input alignment, blue is accurately estimated output alignment, and green lines indicates matches using learned descriptors.

has that for each such point x in I, the corresponding point $x' \in I'$ is calculated by

$$x' = \mathbf{w}(x; \psi). \tag{5.1}$$

Sll implementations and evaluations discussed in this section are limited to $\mathbf{w}(\cdot; \psi)$ defined by a homography transformation, while the algorithm discussed can be applied to any transformation defined by form $\mathbf{w}(\cdot; \psi)$. For general image pairs which are taken at non-trivial 3D scenes, such transformation does not exist, and these cases will be discussed in next section.

5.3.1 Preliminaries

We first briefly review how local descriptors are learned using siamese networks and how these networks are trained. The neural network takes square patches as input and outputs a feature descriptor vector. We denote $n \times n$ pixel patches taken from a c channel image as $\mathbf{x} \in \mathbb{R}^{n \times n \times c}$. The network outputs a d–dimensional vector that is denoted as $\boldsymbol{f}(\mathbf{x}; \theta) \in \mathbb{R}^d$. Here, θ is a vector denoting the network parameters (or weights). We sometimes use

$\boldsymbol{f}(\mathbf{x})$ instead of $\boldsymbol{f}(\mathbf{x};\theta)$ for notational brevity.

The training data consists of two sets of pairs of patches. Each patch pair is denoted by $\{\mathbf{x},\mathbf{x}'\}$. The first set denoted by $\mathcal{P}$ contains true correspondences i.e. pairs of patches from different images that depict the same scene point or object. The negative set denoted by $\mathcal{N}$ contains pairs of patches that are not corresponding and depict different scene points and objects. Training the network involves learning an embedding, where $\|\boldsymbol{f}(\mathbf{x}) - \boldsymbol{f}(\mathbf{x}')\|_2$ is small for positive pairs $\{\mathbf{x},\mathbf{x}'\} \in \mathcal{P}$, and the distance is large for negative pairs $\{\mathbf{x},\mathbf{x}'\} \in \mathcal{N}$. The loss function used during training is called contrastive loss [123, 124, 125, 138]. It is defined for input pairs and has the following form.

$$\begin{aligned} \boldsymbol{L}_0(\mathbf{x},\mathbf{x}';\theta) &= \|\boldsymbol{f}(\mathbf{x};\theta) - \boldsymbol{f}(\mathbf{x}';\theta)\|_2 \\ \boldsymbol{L}_1(\mathbf{x},\mathbf{x}';\theta) &= \max(\,0,\,\mu - \|\boldsymbol{f}(\mathbf{x};\theta) - \boldsymbol{f}(\mathbf{x}';\theta)\|_2) \end{aligned} \tag{5.2}$$

The hyperparameter μ denotes a margin and is often set to the value 1. The loss function $\boldsymbol{L}_0$ encourages pairs of vectors to have smaller pairwise distances. In contrast, the hinge loss $\boldsymbol{L}_1$ encourages the pairwise distances to increase and imposes a penalty when those distances become smaller than the margin μ. The network parameters θ are computed by solving the following optimization problem.

$$\underset{\theta}{\operatorname{argmin}} \Big(\sum_{i=1}^{|\mathcal{P}|} \boldsymbol{L}_0(\mathbf{x}_i,\mathbf{x}_i';\theta) + \sum_{j=1}^{|\mathcal{N}|} \boldsymbol{L}_1(\mathbf{x}_j,\mathbf{x}_j';\theta) \Big) \tag{5.3}$$

Stochastic gradient descent (SGD) is often used for this problem and cross-validation is used to pick the best model.

5.3.2 Algorithm

Given an image pair $\{I, I'\}$, we assume that we know the family of 2D warping functions $\mathbf{w}(\cdot\,;\psi) : \mathbb{R}^2 \to \mathbb{R}^2$ with parameter vector ψ that transforms a pixel location or a patch in I to the corresponding location or patch in I'. We will now present our method to estimate

the parameters ψ while simultaneously computing the network parameters θ. The resulting embedding $\boldsymbol{f}(\mathbf{x};\theta)$ will be specific to I and I'.

Before describing our weakly supervised method is full detail, we first discuss a simpler learning problem, that of learning θ when the parameter ψ is known. This problem can be solved using supervised learning. We then motivate how this method can be extended to the weakly supervised setting and finally present the proposed approach to learn ψ and θ jointly using a joint optimization framework. Finally, we also present a hybrid variant of siamese and pseudo siamese networks which is useful in cases where the input image modalities differ a lot e.g. RGB and NIR.

Descriptor learning from an aligned image pair

When the alignment parameters ψ for an image pair are known, we can use the warping function to extract corresponding patches from the images to train $\boldsymbol{f}(\mathbf{x};\theta)$. We first select a set of image patches in the first image I. For each such patch $\mathbf{x}$ in I, we find the warped patch $\mathbf{w}(\mathbf{x};\psi)$ in I' and insert them into the set of positive pairs $\mathcal{P}$. Similarly, we construct the set of negative pairs $\mathcal{N}$ by randomly sampling patches in I' whose location in the image are at least τ pixels away from the location of the true corresponding patch. The model can now be trained by minimizing a pairwise loss function, similar to the one in Equation 5.2.

$$\begin{aligned}\boldsymbol{L}_0(\mathbf{x};\psi,\theta) &= \|\boldsymbol{f}(\mathbf{x};\theta) - \boldsymbol{f}(\mathbf{w}(\mathbf{x};\psi);\theta)\|_2 \\ \boldsymbol{L}_1(\mathbf{x},\mathbf{x}';\theta) &= \max(\,0,\,\mu - \|\boldsymbol{f}(\mathbf{x};\theta) - \boldsymbol{f}(\mathbf{x}';\theta)\|_2)\end{aligned} \tag{5.4}$$

The network parameters are learned by minimizing the following objective over the pairs in $\mathcal{P}$ and $\mathcal{N}$.

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \Big(\sum_{i=1}^{|\mathcal{P}|} \boldsymbol{L}_0(\mathbf{x}_i;\psi,\theta) + \sum_{j=1}^{|\mathcal{N}|} \boldsymbol{L}_1(\mathbf{x}_j,\mathbf{x}'_j;\theta) \Big) \tag{5.5}$$

This method still uses supervised learning but does so in an uncommon setting. The embedding $\boldsymbol{f}(\mathbf{x};\theta)$ is being learned from only a single image pair. Hence, the training set is

small and the model is very likely to overfit to the data. When the alignment parameter ψ is accurate, the embedding is still meaningful for this image. We will now analyze what happens when the alignment is inaccurate and the correspondences it induces are imprecise.

From supervised to weakly supervised

To simulate the effect of imprecise alignment on an image pair, we add x and y translational offsets to the true warping function to generate several imprecise candidates. For each alignment candidate, we applied the supervised method just described previously to learn a different embedding $\boldsymbol{f}(\mathbf{x};\theta)$ from scratch and recorded the loss obtained at the end of training. Figure 5.2 shows a visualization of the training loss for all x and y offsets up to $\pm$ 100 pixels.

The visualization shows the effect of misalignment on the training loss. There is a well defined minimum at the center, i.e. when there is no misalignment. More over, the cost surface is smooth and has a stable gradient near the center and does not have any significant spurious local minimal. This observation motivated the question. *Can we minimize the same pairwise loss to also iteratively estimate the alignment ψ as we train the neural network?*

This leads to our *self-supervised* method to jointly learn the descriptor and the warping parameters. We still use the pairwise loss function defined in Eq. 5.4, but the warping parameter ψ is no longer assumed to be known. Instead it is a variable in the loss function optimization.

$$\theta^*, \psi^* = \underset{\theta,\psi}{\operatorname{argmin}} \Big(\sum_{i=1}^{|\mathcal{P}|} \boldsymbol{L}_0(\mathbf{x}_i;\psi,\theta) + \sum_{j=1}^{|\mathcal{N}|} \boldsymbol{L}_1(\mathbf{x}_j,\mathbf{x}'_j;\theta) \Big) \tag{5.6}$$

Our self-supervised method does not need any transformation supervision, but the training loss will in general be non-convex. With iterative optimization techniques based on gradient descent, there is no guarantee of convergence to the correct estimate of ψ start-

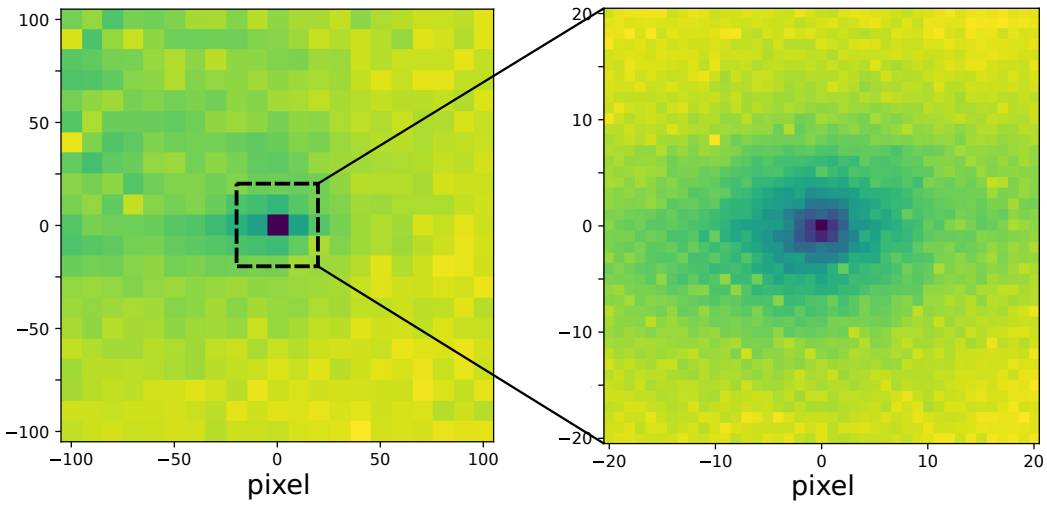


Figure 5.2: Visualization of final training losses (see Eq. 5.4) for supervised method on an image pair, where 2D translations (up to $\pm$ 100 pixels) were added to the true locations in the second image. Darker color indicates lower loss.

ing from an arbitrary initial value. Thus, our method is *weakly supervised*. It needs a reasonable initialization of ψ or requires the input images to be roughly aligned. While this sounds like a limitation, coarse alignment is available in many cases and our empirical evidence shows that our method can handle a fairly large amount of misalignment.

Joint alignment and weakly supervised learning

In all the implementations and evaluations of this section, I assume that the warping model $\mathbf{w}$ is based on a 2D homography. Thus, we can handle image pairs from a purely zooming or rotating camera or overlapping images of a planar scene from one or more cameras.

We now describe the joint optimization for the homography case in more details. To generate patches from an image, we extract multiple randomly sampled 2D keypoints from the image. Each keypoint $\mathbf{p}$ has a position (x, y), orientation ϕ and scale s, from which we can resample a $n \times n$ square patch using bilinear interpolation. Mathematically, we write this as $\mathbf{x} = \mathrm{B}(\mathbf{p}; I)$. To obtain corresponding patches, we transform the keypoint $\mathbf{p}$ in the image I using the homography to obtain the transformed keypoint $\mathbf{p}'$ in I' and then

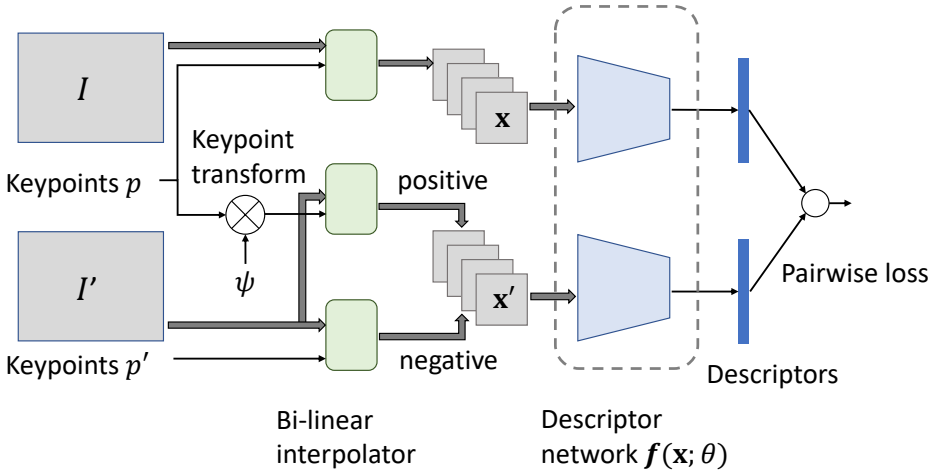


Figure 5.3: Overview of joint model for descriptor learning and estimating alignment.

resample the patch associated with $\mathbf{p}'$. Here $\mathbf{w}_k$ is the same warping function as $\mathbf{w}$, but $\mathbf{w}_k$ transforms keypoints instead of patches. Mathematically, we have,

$$\mathbf{x} = \mathrm{B}(\mathbf{p}; I), \;\; \mathbf{w}(\mathbf{x}; \psi) = \mathrm{B}(\mathbf{p}'; I') = \mathrm{B}(\mathbf{w}_k(\mathbf{p}; \psi); I'). \tag{5.7}$$

Bilinear interpolation is differentiable with respect to the homography parameters. When we substitute $\mathbf{w}(\mathbf{x}; \psi)$ from Eq. 5.7 into Eqs. 5.4 and 5.6, the new training loss remains differentiable with respect to θ and ψ. Thus, we can use backpropagation to compute the derivatives. Figure 5.3 illustrates our model. Since the parameter vector ψ gets iteratively updated, we compute the positive pairs using the updated homography from scratch and regenerate the positive training set on-the-fly in each iteration. However, the set of negative pairs remains fixed throughout. The neural network architecture, implementation details of keypoint warping and the homography parametrization are discussed in Section 5.3.3.

Partially shared pseudo siamese network

Since siamese networks use shared weights for both input patches, they may perform poorly when the input image pairs have different modalities, e.g. RGB and NIR, since one set of parameters may not capture the statistics of both modalities. Similar ideas for unsharing

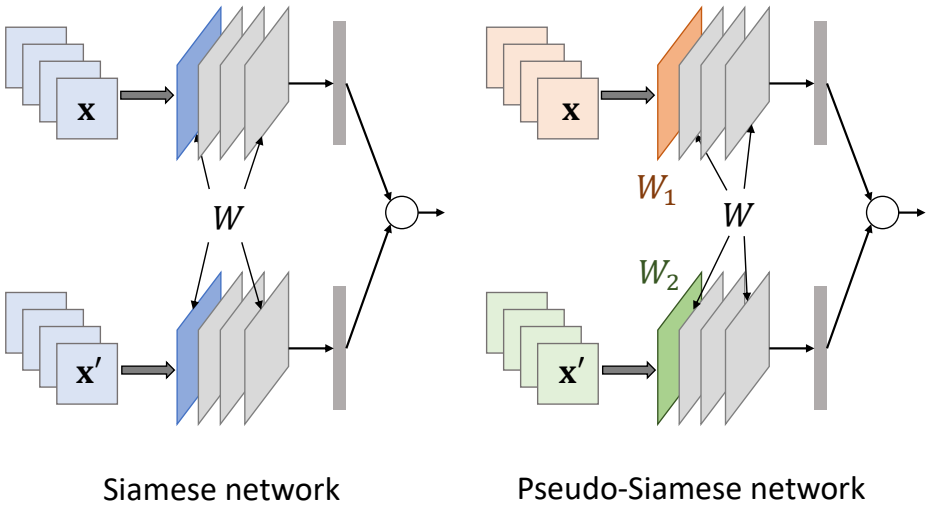


Figure 5.4: We train standard siamese networks and a special form of pseudo-siamese networks with unshared weights in the first layer to adapt to different image modalities (shown by orange and green).

network weights have been investigated in domain adaptation and transfer learning [139, 140]. In *pseudo-siamese* networks, both networks have the same structure but have different copies of the weights. They have been used for matching different modalities [141, 142]. but do not provide a significant gain when both inputs have the same modality [143]. With twice the parameters, pseudo-siamese networks tend to perform worse than siamese networks when training data is limited.

We explore a special case of pseudo-siameses networks where the weights of only the first layer of both networks are different i.e. the first layer is unshared, but the remaining parameters of the network are shared (see Fig. 5.4). This allows the first layer to adapt to the different modalities but only adds a small number of parameters to the model. This is important for us since the training sets are quite small.

5.3.3 Implementation details

Homography parameterization for SGD

Unlike second-order methods e.g. Gauss-Newton, first-order methods like SGD are not curvature-aware and have trouble optimizing ill-conditioned problems when different dimensions of the parameter space differ a lot in scale. The usual 8-DOF parameterization for the homography H has eight parameters often have very different scales.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}. \tag{5.8}$$

Fig. 5.5 shows the scale variation of the eight parameters for some homographies in the HPatches dataset [144].

Therefore, we choose a well-conditioned homography parametrization that is more suitable for use with SGD. Based on *dimensional analysis* [145] in physics, we check the *unit* of H the homography mapping although, there are no physical units.

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1}, y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1}. \tag{5.9}$$

Here, x, y, x' and y' are in pixels. So to reach dimensional homogeneity, $h_{11}, h_{12}, h_{21}, h_{22}$ must have *unit* 1, h_{13}, h_{23} must have pixel *unit*, and h_{31}, h_{32} must have pixel^{-1} *unit*. Thus, we normalize H using the image dimension $w \times h$ and this gives the following 8-dimensional vector for ψ.

$$\alpha\big[h_{11} - 1, h_{12}, h_{21}, h_{22} - 1, \frac{h_{13}}{w}, \frac{h_{23}}{h}, h_{31}w, h_{32}h\big] \tag{5.10}$$

Here, α is a scale hyperparameter that we set to 64. This parametrization is scale normalized (see Fig. 5.5) and conveniently maps $\psi = 0$ to the identity homography.

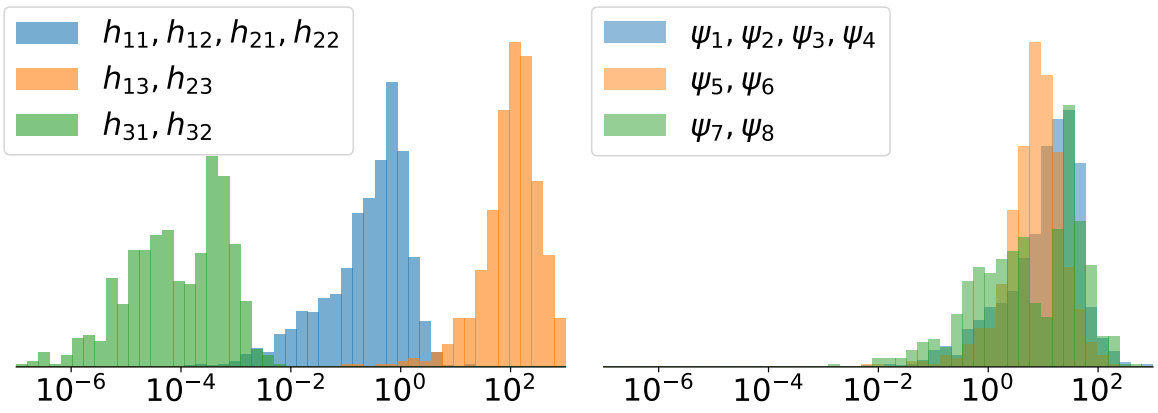


Figure 5.5: Homography parameter's distributions of dataset [144] in log-scale, left is before normalization in H, and right is normalization in ψ with $\alpha = 64$.

Keypoint transformation under homography

Here we define how an image keypoint can be warped through a homography transformation. The position (x', y') of a warped keypoint $\mathbf{p}'$ is given by Eq. 5.9. To find the orientation ϕ' and scale s' of the point $\mathbf{p}'$, we treat $\mathbf{p}$ as a 2D vector $[v_0, v_1]^T$ with orientation ϕ and length s with origin (x, y) in image I. Then, we have

$$v_0 \propto s\cos\phi, \qquad\qquad v_1 \propto s\sin\phi \tag{5.11}$$

Warping a keypoint is the same as finding the warped vector $(v_0', v_1')^T$ in I'. Assuming the vectors are short, we can use finite differences on Eq. 5.9 to compute the values of

$$v_0' \approx \frac{h_{11}v_0 + h_{12}v_1}{h_{31}x + h_{32}y + 1}, v_1' \approx \frac{h_{21}v_0 + h_{22}v_1}{h_{31}x + h_{32}y + 1} \tag{5.12}$$

The orientation and scale of $\mathbf{p}'$ can be computed by substituting values of v_0' and v_1' from Eq. 5.12 into these equations.

$$\phi' = \arctan\left(\frac{v_1'}{v_0'}\right), s' = \sqrt{(v_0')^2 + (v_1')^2} \tag{5.13}$$

CNN training

Implementation details about how keypoint sets $\{\mathbf{p}\}$ are selected are described below. To obtain image patches to train our model, we select keypoint sets $\{\mathbf{p}\}$ and $\{\mathbf{p}'\}$ with sufficient randomization to reduce the effect of overfitting. We sample approximately 4000 keypoints in each image. The positions of these keypoints are sampled from areas with sufficient contrast by selecting areas were the local gradient magnitude exceeds 0.05[1] and selecting pixels randomly from a uniform distribution. The orientation is sampled uniformly from the range $[0, 2\pi]$ and the scale values are sampled such that $\log_2 s$ is distributed uniformly in the interval $[0, 4]$. We also check that the warped keypoint $\mathbf{p}'$ lies inside the boundary of image I'. Although the cropped patches extracted from $\{\mathbf{p}\}$ and $\{\mathbf{p}'\}$ spatially overlap, selecting the orientation and scales randomly provides effective data augmentation.

We use a shallow network architecture: `Conv(16,32,5,1)-Tanh-MaxPool(12,-,2,2)-Conv(6,64,3,1)-Tanh-FC(256)`. The parameters of each layer are denoted by `(Input,Channel,Kernel,Stride)`. The patch size is 16×16 pixels and descriptor dimension is $d = 256$. We chose this architecture because similar shallow networks have been shown to work well if trained properly [125, 123]. Second, due to limited training data we prefer a lower network capacity to reduce the risk of overfitting. However, our proposed ideas can be combined with other network architectures as well. The optimization is done using SGD with batch size of 64 and momentum of 0.9, with an initial learning rate of 10^{-4} which is temporally annealed. Our implementation is based on Tensorflow [146].

To address the fact that a large initial misalignment may be present, the training is done in a coarse-to-fine fashion on an image pyramid, where the coarsest level has about 80 pixels in the larger axis. The joint alignment and network training starts at coarsest pyramid level after which the estimated parameter ψ^* is used to initialize ψ at the next

[1] image intensities have mean 0 and standard deviation 1

level, whereas the network parameters θ^* are discarded. The parameters ψ^* and θ^* from the finest level are the final output.

5.3.4 Evaluation on the HPatches dataset

The proposed method is first evaluated on matching color images from HPatches [144] to compare with existing methods, then the method is tested for aligning aerial RGB and NIR images that were captured using an off the shelf drone. On both datasets, ours descriptor is quantitatively evaluated to SIFT [84] and DAISY [147]. We also compare to four learned approaches – DeepCompare [143] with one (-s) and two stream (-s2s) siamese networks, DeepDesc [123] and DeepPatchMatch [148] which are currently amongst the top ranked methods on HPatches and were trained on the MVSCorr dataset [149]. *Mean average precision (mAP)* is the used metric to evaluate descriptor performance, which is defined by the following: For a single test image pair in both datasets, there is a set of N patches $\{\mathbf{x}_i\}$ extracted from I and corresponding $\{\mathbf{x}'_i\}$ from I'. We first calculate descriptors $\mathbf{d}_i$ and $\mathbf{d}'_i$ for all patches, and for each descriptor $\mathbf{d}_i$ we look for L^2 nearest neighbor in $\{\mathbf{d}'_i\}$ with result nearest neighbor index k_i. The average precision (AP) is the percentage of correct nearest neighbors of $\{\mathbf{d}_i\}$ found in $\{\mathbf{d}'_i\}$, defined by

$$AP = \frac{\sum_{i=1}^{N}[i \stackrel{?}{=} k_i]}{N}, \tag{5.14}$$

and the mean average precision (mAP) is defined by average AP across all the image pairs in the dataset.

HPatches [144] contains image pairs rated at EASY, HARD and TOUGH difficulty levels for 116 scenes where 59 scenes have viewpoint variations and the rest vary in illumination. The evaluation protocol in HPatches involves calculating mAP on pre-extracted pairs of grayscale patches. We implement this protocol but using our own patches so that descriptors computed from RGB patches can also be evaluated. To generate the evaluation

patches, we extracted about 1400 DoG keypoints [84] in the first image, warped the underlying patches by the ground truth homographies and sampled the corresponding patches in the second image.

Descriptor performance evaluation

We evaluate four variants of our method for grayscale and RGB patches combined with the siamese network (**Ours-s**) and the pseudo siamese network (**Ours-ps**) respectively. The descriptor mAP evaluation is summarized in Table 5.1, and we show two example image pairs with alignment and descriptor results of **Ours-ps** in Fig. 5.6 and Fig. 5.7. All six baselines were based on grayscale patches. The coarse alignment needed by our method for initialization was simulated by adding random 2D translational perturbations to the true homography, where the shift is equal to 5% of the image size (see later for results with larger shifts).

In this evaluation, **Ours-s-Grayscale** is ranked lower than pre-trained CNN but the gap in mAP is small – 0.57 (ours) v.s. 0.59–0.62 [143, 123, 148]. This is expected, since those CNNs were trained offline on a massive patch dataset whereas in our case, the network is optimized from scratch on each image pair. SIFT and DAISY have much lower mAP of 0.39 and 0.47 respectively. However, our RGB networks have the highest overall accuracy. The pseudo siamese (mAP = 0.633) and the siamese network (mAP = 0.631) both had similar accuracies. The siamese variant **Ours-s-RGB** has the highest mAP in four out of the six subgroups. This evaluation shows that the descriptors learned from scratch by our method are representative even though we do not expect them to generalize to new images.

Analyzing robustness to initial alignment error

We also analyze the robustness of our method to increasing amounts of error in the initial alignment. Figure 5.8 shows the results of these experiments. The plots show both the homography estimation error and the mAP scores for models trained on the six subgroups

Table 5.1: HPatches descriptor evaluation: mAP for six baselines and our variants of our method. For each of the six subgroups, the best method using grayscale patches is highlighted in bold. When one of the RGB methods has a higher mAP than the best grayscale method, it is also highlighted.

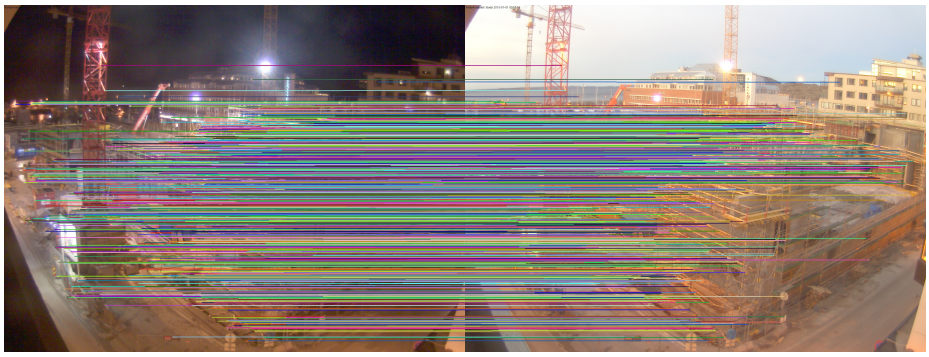
Colorspace	Method	Viewpoint			Illumination			Average
		EASY	HARD	TOUGH	EASY	HARD	TOUGH	
Grayscale	SIFT [84]	0.485	0.297	0.192	0.549	0.431	0.372	0.388
	DAISY [147]	0.683	0.491	0.336	0.579	0.416	0.327	0.472
	DeepCompare-s [143]	0.781	0.542	0.361	0.779	0.632	0.532	0.605
	DeepCompare-s2s [143]	**0.803**	**0.581**	**0.396**	0.776	0.631	0.542	**0.622**
	DeepDesc [123]	0.781	0.554	0.360	**0.792**	**0.655**	0.547	0.615
	DeepPatchMatch [148]	0.770	0.541	0.359	0.760	0.610	0.509	0.592
	Ours-s-Grayscale	0.700	0.426	0.309	0.790	0.644	**0.549**	0.570
	Ours-ps-Grayscale	0.729	0.446	0.290	0.760	0.624	0.520	0.562
RGB	**Ours-s-RGB**	0.763	0.525	**0.397**	**0.813**	**0.679**	**0.607**	0.631
	Ours-ps-RGB	0.781	0.563	0.388	0.808	0.664	0.592	**0.633**

(a) Alignment Result

(b) Matching Result

Figure 5.6: TOUGH pair from HPatches (v_coffeehouse): (a) LEFT: The two input images. MIDDLE: The result in the upper row is the initial alignment. Here both images have been overlaid and blended with each other. The lower row shows the alignment computed by our estimated homography. RIGHT: Zoomed-in views of a specific region of the image shows the quality of the initial and final alignments. (b) The sparse feature matches between the two images. These are computed by extracting SIFT keypoints and matching feature descriptors learned using our method. The matches are filtered using a geometric verification step that robustly fits a homography and finds inlier matches. The outliers are not shown.

(a) Alignment result



(b) Matching result

Figure 5.7: TOUGH pair from HPatches (i_contruction): (a) LEFT: The two input images. MIDDLE: The result in the upper row is the initial alignment. Here both images have been overlaid and blended with each other. The lower row shows the alignment computed by our estimated homography. RIGHT: Zoomed-in views of a specific region of the image shows the quality of the initial and final alignments. (b) The sparse feature matches between the two images. These are computed by extracting SIFT keypoints and matching feature descriptors learned using our method. The matches are filtered using a geometric verification step that robustly fits a homography and finds inlier matches. The outliers are not shown.

starting with a different amount of translational perturbation. The homography error is equal to the average warping error of the true feature matches under the estimated homography, after normalizing the error by the image size. The plots show that the homography error is very low for small perturbations (up to 7.5%) which indicates accurate alignment. The alignment error does increase with higher perturbation. However, it is worth noting that here, we adhere to the standard descriptor evaluation protocol and avoid RANSAC and geometric constraints to *robustly* estimate the homography which would have easily yielded more accurate alignments.

For perturbations between 0.0–0.1, the mAP curves are mostly flat i.e. the mAP drop is fairly small. The mAP drops by an amount between 0.01 and 0.04 for the six subgroups. In these plots, a perturbation of 0.1 is equal to a 2D shift of 10% of the image size. As the perturbation increases, the accuracy of our method drops gradually. At 0.2, the mAP is still reasonably high for the EASY and HARD groups whereas performance drops more for the TOUGH group.

5.3.5 Evaluation on RGB and NIR images

Next, we report our evaluation of RGB–NIR image alignment which has useful applications in precision agriculture (see example in Fig. 5.9). The main difficulties here are frequent gradient reversal and the lack of correlation in the visible spectrum and the NIR band (770–810nm).

We collected 50 RGB-NIR image pairs and manually annotated sparse correspondences in them and then recovered ground truth homographies. We split these into two sets – a training set of 40 pairs and a test set of 10 pairs. We evaluate the descriptor performance and homography estimation accuracy on the training set, which we present next.

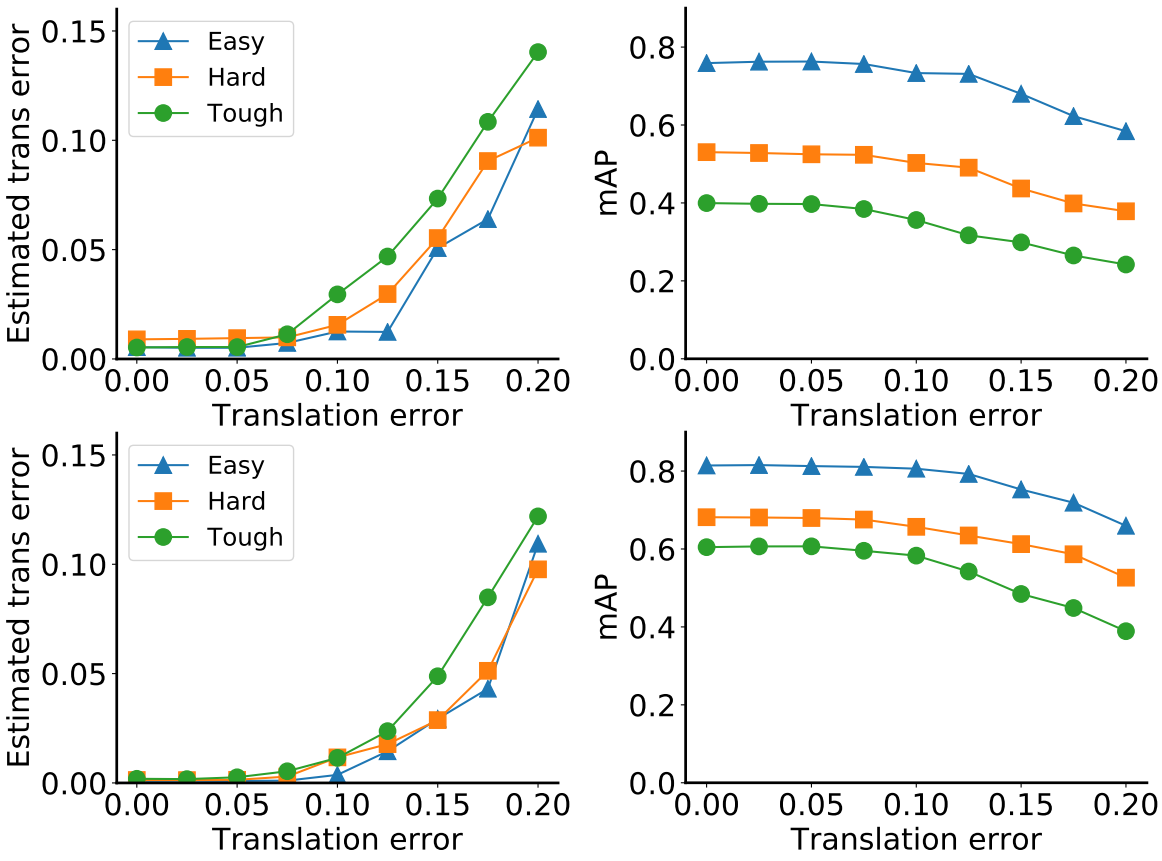


Figure 5.8: The average error of the estimated homography and mAP score of our method on EASY, HARD and TOUGH pairs for a range of initial translation errors (expressed as a fraction of the image size). Upper and lower rows are for "v" and "i" groups respectively in HPatches [144] (see text).

Descriptor performance evaluation

We calculate the mAP of our descriptors on the 40 training pairs using the same protocol used for HPatches. In each case, an identity transformation was used for initialization. We compared our method with the six baselines where the color patches were converted to grayscale. Table. 5.2 summarizes the descriptor performance evaluation for this case. Our method performs significantly outperforms all existing methods. The existing learned methods were not trained for these modalities and are not applicable when datasets with ground truth correspondences are unavailable. In this case, the pseudo-siamese network (mAP = 0.603) is much more accurate than the siamese network (mAP = 0.404) for the

Table 5.2: Descriptor evaluation on RGB and NIR image alignment, shows the descriptor performance from our weakly supervised method on the 40 training pairs which clearly outperforms all baselines.

Method	RGB vs. NIR on train set(40)
SIFT	0.098
DAISY	0.030
DeepCompare-s	0.068
DeepCompare-s2s	0.101
DeepDesc	0.070
DeepPatchMatch	0.089
Ours-s	0.404
Ours-ps	**0.603**

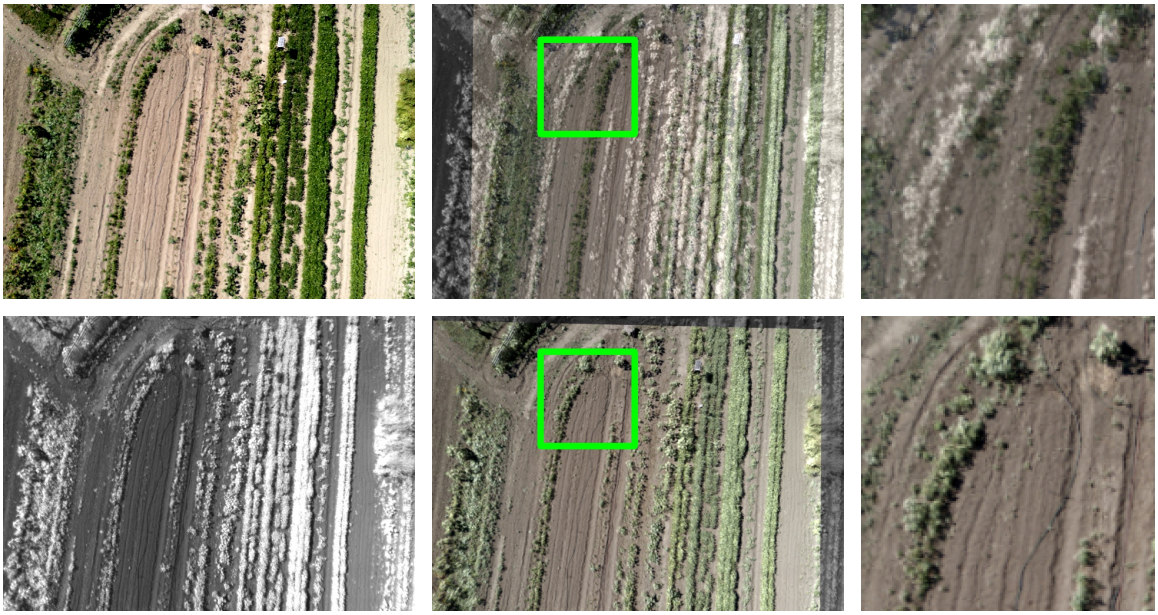
reason discussed earlier.

Analyzing robustness to initial alignment error

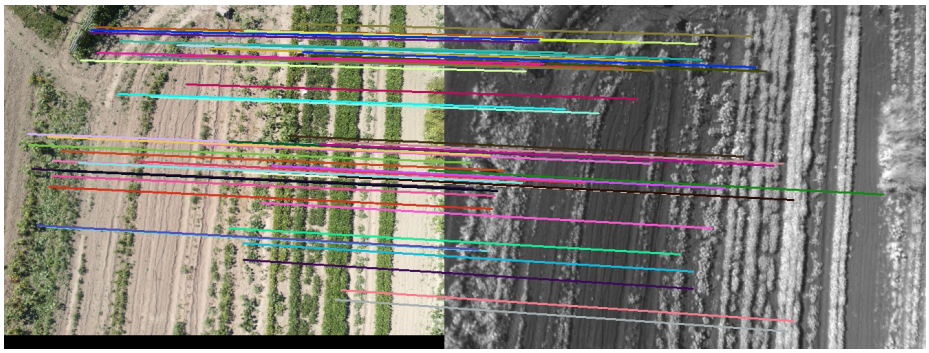
Once again, we evaluate our method by adding translational shifts to the ground truth alignment and simulate initializations of increasing difficulty. Since mutual information is sometimes used to align images of different modalities [150], we compare our pseudo-siamese network to an advanced mutual information based image registration method – `Elastix` [41] which uses Mattes' mutual information metric [151], with the same coarse to fine pyramid resolution as our method. The results are shown in Fig. 5.10. With increasing translational perturbation, the mAP decreases gradually and the homography error increases as expected. However, our method is consistently more reliable than `Elastix`. In particular, our method has near-zero homography error (i.e. all pairs in training set are accurately aligned) up to a perturbation of 0.05 whereas `Elastix` is accurate only up to a much smaller perturbation of 0.025.

Automatic descriptor learning

We now evaluate the effectiveness of our method for automatically building a dataset without human annotation. We train a supervised descriptor on such a dataset and test the



(a) Alignment result



(b) Matching result

Figure 5.9: RGB NIR image pair 1: (a) LEFT: The two input images. MIDDLE: The result in the upper row is the initial alignment. Here both images have been overlaid and blended with each other. The lower row shows the alignment computed by our estimated homography. RIGHT: Zoomed-in views of a specific region of the image shows the quality of the initial and final alignments. (b) The sparse feature matches between the two images. These are computed by extracting SIFT keypoints and matching feature descriptors learned using our method. The matches are filtered using a geometric verification step that robustly fits a homography and finds inlier matches. The outliers are not shown.

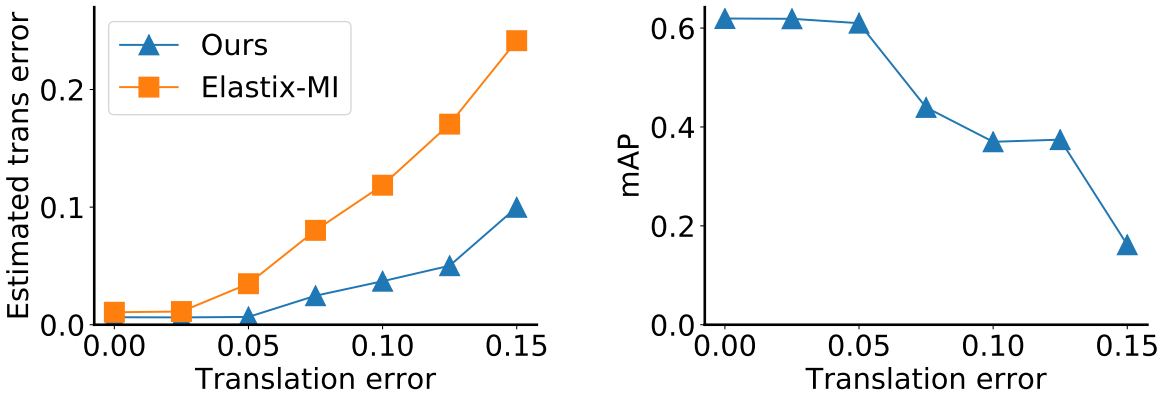


Figure 5.10: Homography errors (alignment accuracy) and mAP (descriptor performance) for increasing error levels in the initial alignment. Results are presented for both our method and `Elastix` [41] which uses mutual information for image registration. Our method consistently handles a larger degree of initial misalignment.

descriptor performance. We run our method on 40 RGB/NIR image pairs. Using the homography obtained from each pair we extract correspondences. Specifically, DoG features points extracted from the RGB images are warped by the estimated homography. We construct a set of 62K positive pairs in the training set. The supervised descriptor is a siamese CNN architecture (see Section 5.3.3), and trained using the same hyperparameters described earlier except that fewer training epochs were used.

This supervised descriptor network is evaluated using the same protocol as before but on the 10 test image pairs, Table. 5.3 shows the mAP for this method. The mAP of 0.556 is comparable to the that of our weakly supervised method and significantly higher than any of the existing methods. The reported performance was obtained without tuning network architecture or any of its hyperparameters. These preliminary results are quite promising and shows that our weakly supervised method is indeed feasible for automatically learning local descriptors from coarsely aligned images. Higher accuracy is possible by running this method at scale on multiple scenes and by testing different network architectures and training objectives.

Table 5.3: Descriptor evaluation on RGB and NIR image alignment, shows the descriptor performance on the 10 test pairs. Those descriptors were trained in supervised fashion on a patch dataset constructed automatically using our weakly supervised method.

Method	RGB vs. NIR on test set(10)
SIFT	0.085
DAISY	0.040
DeepCompare-s	0.071
DeepCompare-s2s	0.098
DeepDesc	0.066
DeepPatchMatch	0.080
Automatic-s (supervised)	0.475
Automatic-ps (supervised)	**0.556**

5.4 Extension to 3D cases

In the final part of this chapter I extend the proposed weakly-supervised descriptor and alignment learning to cases captured in general 3D scenes. The algorithm proposed in last section is only able to be applied to the cases which there is a single parameter ψ defines warping between the image pair, which meets $x' = \mathbf{w}(x; \psi)$. But for x and x' from image pair I and I' which are captured at a non-planar 3D scene, there is no such warping function exists. In fact, given x and relative pose between I and I', all possible x' are lay in the epipolar line defined by the relative pose and camera calibrations.

In this section, I propose an extension of the proposed weakly-supervised descriptor learning, which extends the method to align and match image pairs captured in general 3D scenes, with the help from extra 3D information input. The basic idea is that if the depth of x is known, we can back-project the 2D image point to a 3D point, not a line, in this way we can uniquely determine where x' is on epipolar line.

5.4.1 Algorithm

Although a unique warping function $x' = \mathbf{w}(x; \psi)$ for x and x' from image pair I and I' which are captured at a non-planar 3D scene does not exist, since a given x only determine

x' on epipolar line, but with help from extra 3D information, we can determine the unique x'. If the depth measurement of x in I is known by d, the 3D location of x in camera I's frame is determined by $d\mathbf{K}p$, where $\mathbf{K}$ is camera I's intrinsic calibration matrix, and p is homogeneous coordinate of x defined by

$$p = \begin{bmatrix} x \\ 1 \end{bmatrix}. \tag{5.15}$$

If the relative pose between I and I' is determined by rotation matrix $\mathbf{R}$ plus translation $\mathbf{t}$, the 3D location in camera I''s frame is $d\mathbf{R}{\mathbf{K_1}}^{-1}p + \mathbf{t}$, and then x' can be projected by

$$s \begin{bmatrix} x' \\ 1 \end{bmatrix} = p' = \mathbf{K}'(d\mathbf{R}\mathbf{K}^{-1}p + \mathbf{t}), \tag{5.16}$$

where p' is homogeneous coordinate of x' with unimportant arbitrary scale s, and $\mathbf{K}'$ is camera I''s intrinsic calibration.

Given how x' is projected with the knowing d, the warping function $\mathbf{w}$ of x with accompanied depth input d and warping parameter $\mathbf{R}$ and $\mathbf{t}$ is defined by

$$\mathbf{w}(x, d; \mathbf{R}, \mathbf{t}) = x' \tag{5.17}$$

where x' is calculated by Eq. 5.16. This warping function has similar form to original warping function, but with extra depth input. With the well defined warping function of x to x', we apply the same function to image patches from $\mathbf{x}$ to $\mathbf{x}'$, and the original loss function for joint descriptor and alignment learning in Eq. 5.4 can be updated

$$\begin{aligned} \boldsymbol{L}_0(\mathbf{x}, d; \mathbf{R}, \mathbf{t}, \theta) &= \|\boldsymbol{f}(\mathbf{x}; \theta) - \boldsymbol{f}(\mathbf{w}(\mathbf{x}, d; \mathbf{R}, \mathbf{t}); \theta)\|_2 \\ \boldsymbol{L}_1(\mathbf{x}, \mathbf{x}'; \theta) &= \max(\,0,\, \mu - \|\boldsymbol{f}(\mathbf{x}; \theta) - \boldsymbol{f}(\mathbf{x}'; \theta)\|_2), \end{aligned} \tag{5.18}$$

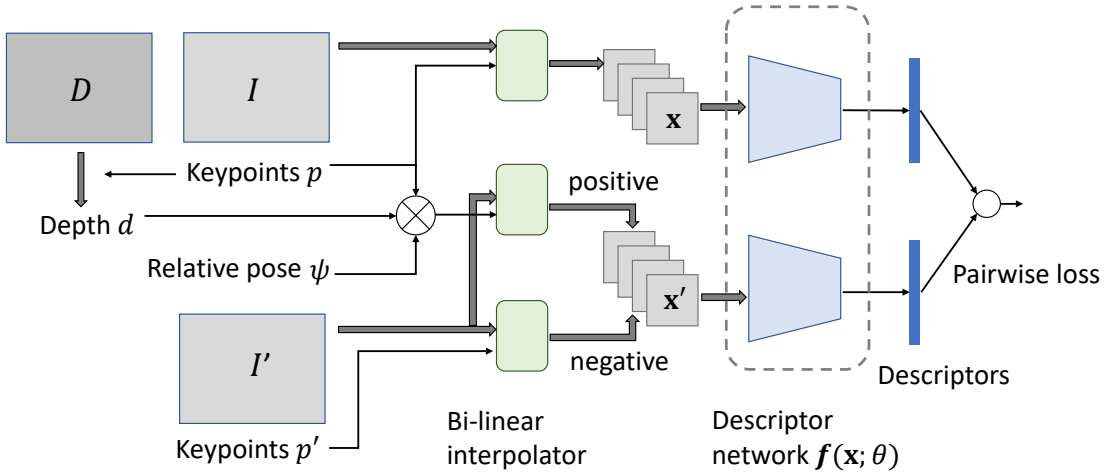


Figure 5.11: Overview of joint model for descriptor learning and estimating relative pose.

and the joint descriptor and alignment learning can be formulated by

$$\theta^*, \mathbf{R}^*, \mathbf{t}^* = \underset{\theta, \mathbf{R}, \mathbf{t}}{\operatorname{argmin}} \Big(\sum_{i=1}^{|\mathcal{P}|} \boldsymbol{L}_0(\mathbf{x}_i, d_i; \mathbf{R}, \mathbf{t}, \theta) + \sum_{j=1}^{|\mathcal{N}|} \boldsymbol{L}_1(\mathbf{x}_j, \mathbf{x}'_j; \theta) \Big). \tag{5.19}$$

Fig. 5.11 shows the diagram of the joint descriptor and alignment learning under 3D. The major change compared to the original method is that the depth image on the left. Other than I and I', depth image D captured at I viewpoint is used as algorithm input, and point depth d is queried at p on D. The source of D can be either direct depth measurements from sensors including LIDAR and Kinect, or from 3D reconstructions including structure from motion and multiview stereo.

5.4.2 Implementation details

Normalized relative pose parameterization

In the proposed joint descriptor and alignment learning under 3D by Eq. 5.19, the objective is to optimize (learn) the neural network weights θ and relative pose $\mathbf{R}$ and $\mathbf{t}$ between I and I' jointly. But as discussed in Sec. 5.3.3, similarly to joint neural network and homography learning, the problem is not well conditioned since parameters to be optimized have

different scale.

Similar to Sec. 5.3.3, inspired by *dimensional analysis* [145] in physics, we check the *unit* of θ, $\mathbf{R}$ and $\mathbf{t}$. Apparently θ and $\mathbf{R}$ have no physical unit, and $\mathbf{t}$ has a length unit (like meter or feet). The relative pose $\mathbf{R}$ and $\mathbf{t}$ are normalized to a single 6-DOF parameterization ψ which is optimized, so joint descriptor and alignment learning first defined in Eq. 5.19 can formulated to a well-conditioned form by

$$\theta^*, \psi^* = \underset{\theta,\psi}{\operatorname{argmin}} \Big(\sum_{i=1}^{|\mathcal{P}|} \boldsymbol{L}_0(\mathbf{x}_i, d_i; \psi, \theta) + \sum_{j=1}^{|\mathcal{N}|} \boldsymbol{L}_1(\mathbf{x}_j, \mathbf{x}'_j; \theta) \Big) \tag{5.20}$$

where ψ is defined by

$$\psi \doteq \alpha \begin{bmatrix} \boldsymbol{\omega} \\ \mathbf{t}/d_{avg} \end{bmatrix}. \tag{5.21}$$

In ψ's definition, d_{avg} is the average depth of depth image D (exclude no measurement area). The reason to use d_{avg} to normalize $\mathbf{t}$ is divided byd_{avg} will remove $\mathbf{t}$'s physical unit, plus a rough estimated scene scale. A simple illustration is if you enlarge a scene by 2, although $\mathbf{t}$ will be doubled, $\mathbf{t}/d_{avg}$ will remain unchanged. α is a hyper-parameter to uniformly scale ψ, to make the joint optimization of ψ and θ well conditioned. In the actual implementation $\alpha = 128$. $\boldsymbol{\omega}$ is the Rodrigues parameterization of the rotation matrix [77]

$$\mathbf{R} = \exp(\boldsymbol{\omega}^\wedge) = \mathbf{I} + \frac{\boldsymbol{\omega}^\wedge}{|\boldsymbol{\omega}|} \sin(|\boldsymbol{\omega}|) + \frac{(\boldsymbol{\omega}^\wedge)^2}{|\boldsymbol{\omega}|^2}(1 - \cos(|\boldsymbol{\omega}|)) \tag{5.22}$$

where $\wedge$ operator constructs a 3×3 skew symmetric matrix from a vector in $\mathbb{R}^3$

$$\boldsymbol{\omega}^\wedge = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}^\wedge = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{5.23}$$

One problem of the above parameterization is that when $\boldsymbol{\omega}$ is close to zero and $\mathbf{R}$ is close

to identity, $|\boldsymbol{\omega}|$ is close to zero, calculating Eq. 5.22 will perform a divide-by-zero. A good approximation is use by when $|\boldsymbol{\omega}|$ is close to zero (in actual implementation when $|\boldsymbol{\omega}| < 10^{-9}$)

$$\exp(\boldsymbol{\omega}^{\wedge}) \approx \mathbf{I} + \boldsymbol{\omega}^{\wedge}, \text{ when } |\boldsymbol{\omega}| \to 0, \tag{5.24}$$

since

$$\frac{\boldsymbol{\omega}^{\wedge}}{|\boldsymbol{\omega}|} \sin(|\boldsymbol{\omega}|) \approx \boldsymbol{\omega}^{\wedge}, \text{ when } |\boldsymbol{\omega}| \to 0. \tag{5.25}$$

CNN training

In my implementation of the 3D enabled weakly-supervised descriptor learning, most implementation details remain the same as for the homography-based method, since the only changed part is the image warping model. The image keypoint and patch sampling strategy remains the same. We still use the shallow network architecture: `Conv(16,32,5,1)-Tanh-MaxPool(12,-,2,2)-Conv(6,64,3,1)-Tanh-FC(256)`, since its lower network capacity makes it fast to train, and reduce the risk of overfitting due to the small amount of training data. The coarse-to-fine fashion on an image pyramid is still used to address possible large initial error on relative pose.

5.4.3 Evaluation on the Middlebury dataset

The evaluation is performed on the Middlebury stereo vision 2014 dataset [152], which includes 23 sets of high resolution multiview image with ground truth depth. While the dataset was designed for benchmarking of multiview stereo methods, since it has multiple image pairs (under different illumination conditioned) with ground truth depth, it is used here to benchmark the proposed descriptor and relative pose learning. Example image sets of Middlebury dataset are shown in Fig. 5.12 and Fig. 5.13. Each multiview image set in Middlebury dataset includes four RGB images and one ground truth depth image: one RGB image I is captured at view one, three RGB images I' are captured at view two

Table 5.4: Middlebury descriptor evaluation: mAP for five baselines and our variants of our method. For each of the subgroups, the best method using grayscale patches is highlighted in bold. When one of the RGB methods has a higher mAP than the best grayscale method, it is also highlighted.

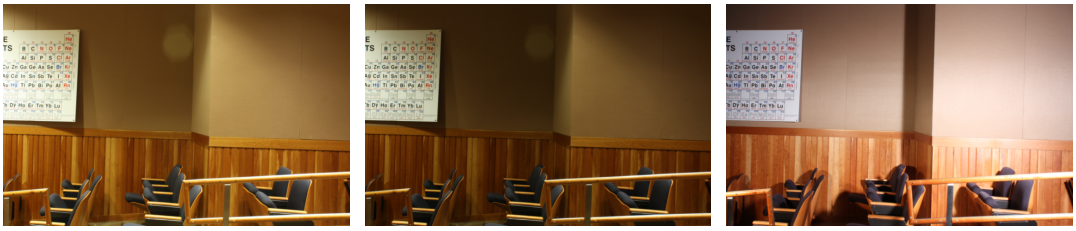
Colorspace	Method	SAME	EXPOSURE	FLASH	Average
Grayscale	SIFT [84]	0.615	0.610	0.390	0.538
	DAISY [147]	0.652	0.513	0.321	0.495
	DeepCompare-s [143]	0.741	0.725	0.495	0.656
	DeepCompare-s2s [143]	0.739	**0.729**	0.492	0.653
	DeepDesc [123]	**0.756**	0.725	**0.504**	**0.662**
	Ours-s-Grayscale	0.524	0.567	0.325	0.472
	Ours-ps-Grayscale	0.492	0.515	0.240	0.415
RGB	**Ours-s-RGB**	0.719	0.685	0.421	0.608
	Ours-ps-RGB	0.660	0.649	0.379	0.563

with three different settings: same exposure with I, different exposure with I, using flash. The resulting three image matching problems between I and I' are designed from easy to hard. Finally the ground truth depth image D is captured at I's viewpoint. *Mean average precision (mAP)* is also the used metric here to evaluate descriptor performance.

Table 5.4 shows the learned descriptor performance in mAP settings, compared to five hand-crafted or learning-based baselines. The results show that the learned descriptor has similar performance compare to the hand-crafted baselines, but poorer performance than the CNN-based baselines. We also realize that the siamese network version (**-s**) of the proposed method, has better performance compared to the pseudo-siamese network version (**-ps**). This is easy to understand, since the task here is matching RGB to RGB image pairs, which are captured at same modality, and only suffers from illumination difference. Although pseudo-siamese network is a better representation to handle illumination difference, due to its larger network capacity, there is possibility that illumination difference is harder to train and have poor performance. Some example image matching results using learned descriptor by siamese network version proposed method are shown in Fig. 5.15 and 5.16.

Compared to the performance of the proposed method with homography image warping

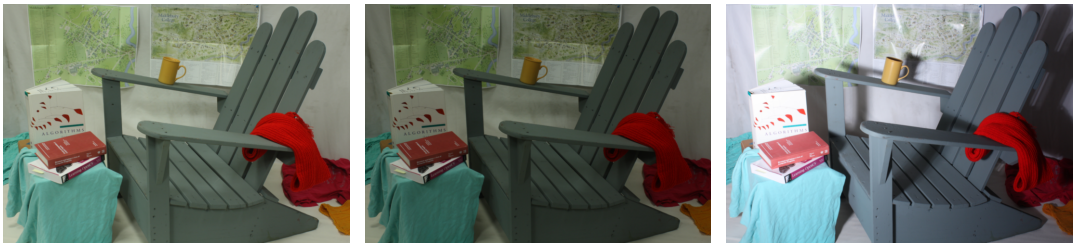
(a) Image I



(b) Image I'

Figure 5.12: Middlebury 2014 dataset example pairs CLASSROOM (a) Left: image I. Middle: depth image D. Right: extracted keypoints in I used for learning. (b) Three I' candidates, captured at same viewpoints (share the same relative pose vs. I), but under different illumination conditions, named as SAME, EXPOSURE and FLASH.

(a) Image I



(b) Image I'

Figure 5.13: Middlebury 2014 dataset example pairs ADIRONDACK (a) Left: image I. Middle: depth image D. Right: extracted keypoints in I used for learning. (b) Three I' candidates, captured at same viewpoints (share the same relative pose vs. I), but under different illumination conditions, named as SAME, EXPOSURE and FLASH.

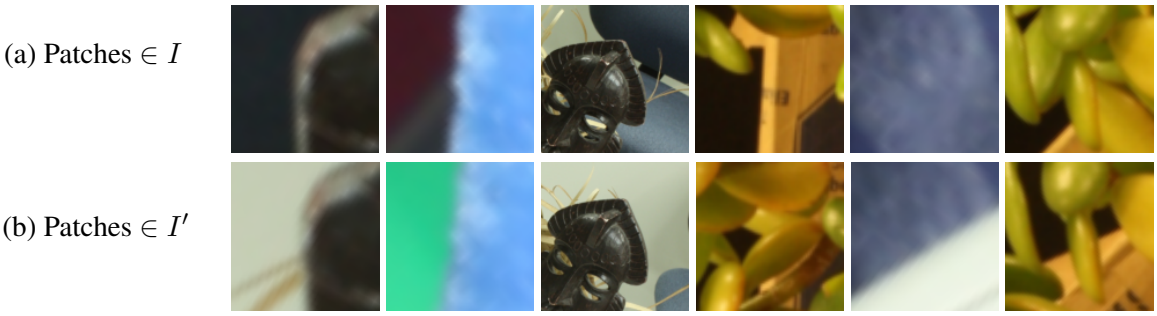
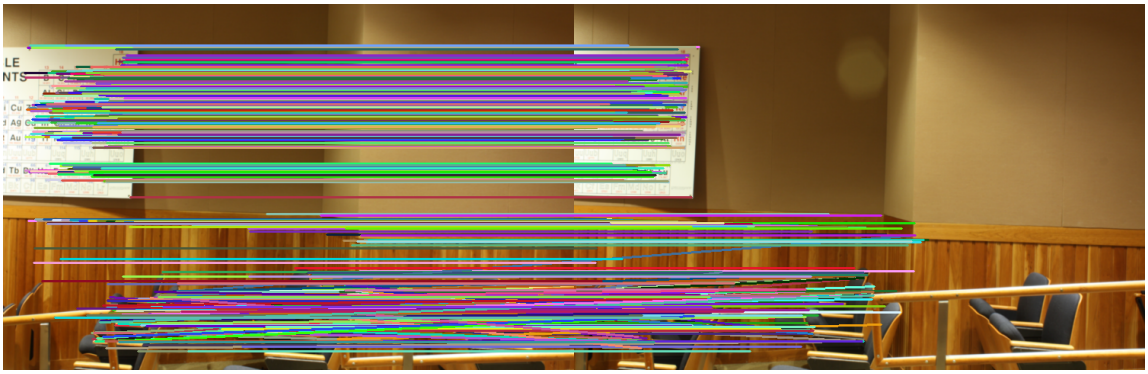


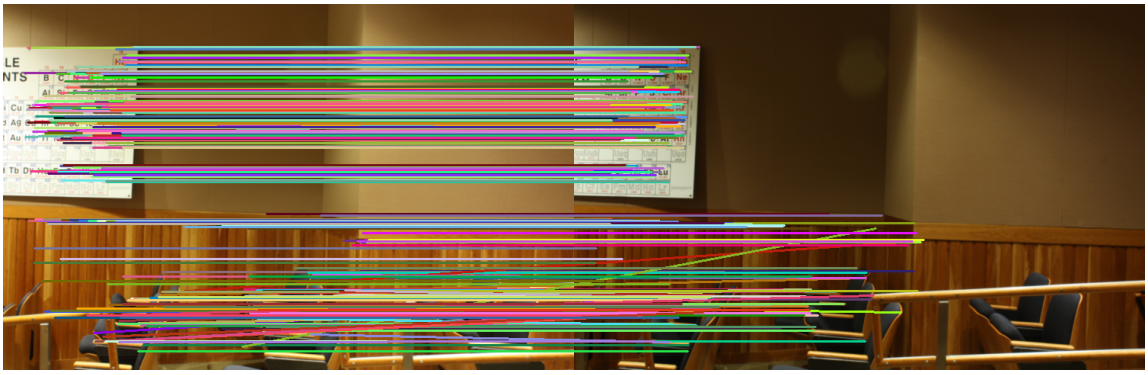
Figure 5.14: Example image patch pairs in various scales, which have significant appearance differences, caused by occlusion in foreground/changes in background due to parallax.

model, the performance of the 3D method is lower. The reasons are twofold: (1) Similar to the issue suffered by the homography-based method, in contrast to other supervised methods, the proposed weakly-supervised method suffers from having only a small amount of training data since they all come from a single image pair, and a limited network representation ability due to the small network capacity. (2) For two images captured for a non-trivial 3D scene at different viewpoints, there is a significant *parallax* effect. This effect changes object appearance by changing its background or by changing its foreground by occlusion. The parallax effect changes appearance of positive patch pairs, makes them have different appearance, although geometrically they are still captured by looking at the same 3D points. The parallax effect does not exist for image pairs under homograhy transformation, and this will makes the learning hard under 3D cases, due to poor quality of positive training data. Some example image pairs suffers parallax effect is shown in Fig. 5.14.

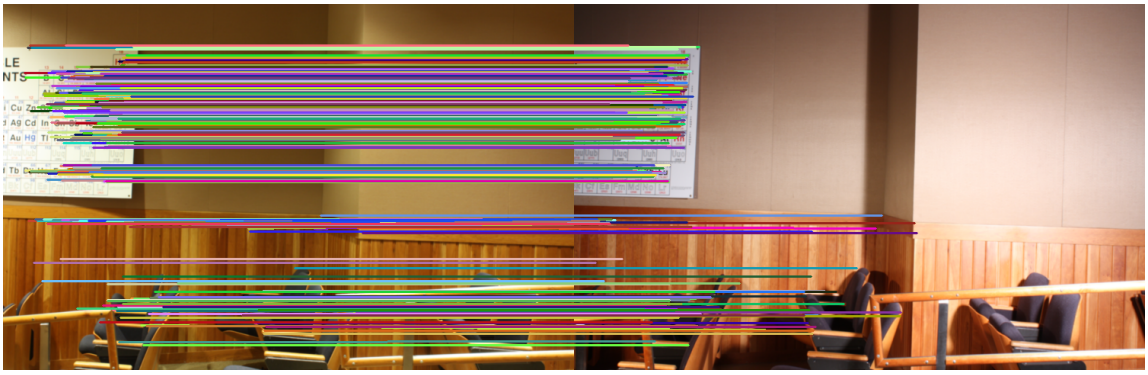
Although the proposed method does not work as well as state-of-the-art supervised learning methods in RGB–RGB image registration tasks, I still believe it has potential for improved performance in cross-modality image registrations, as shown in cases under homography transformation. Unfortunately due to the dataset available, I have not been able perform such tests for now, and leave this as future work.



(a) SAME

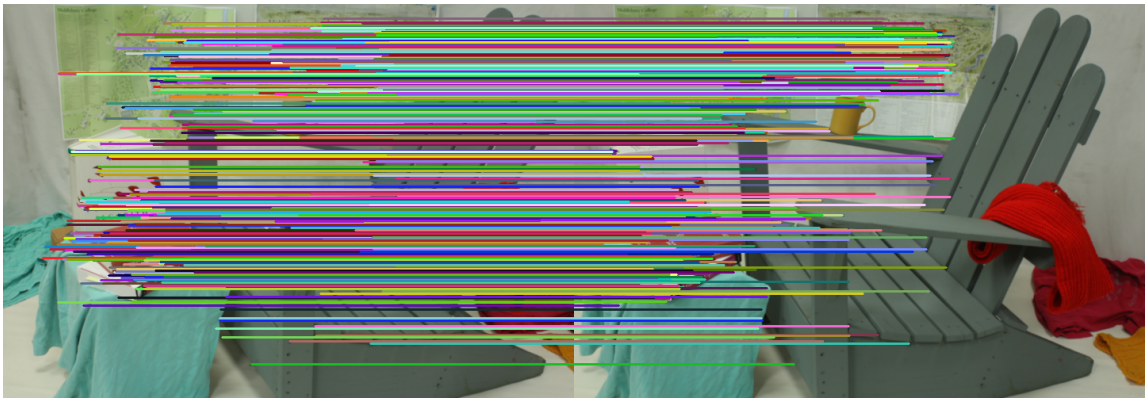


(b) EXPOSURE

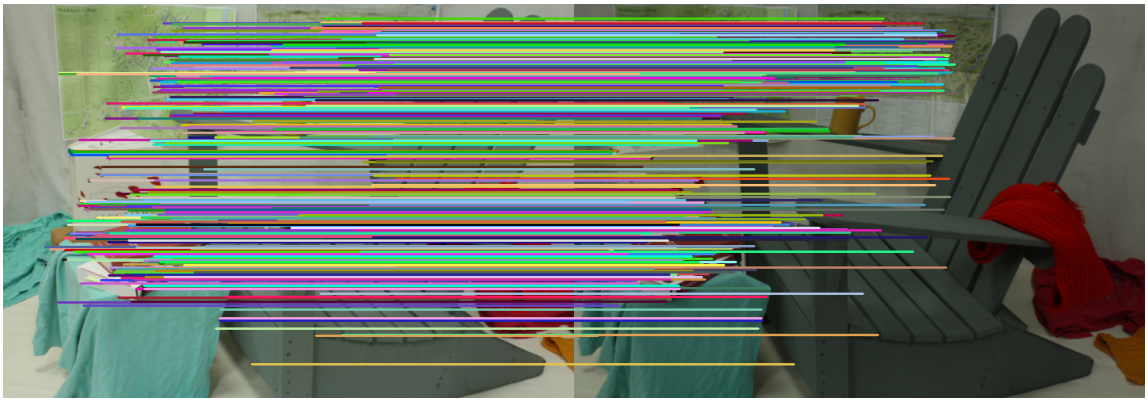


(c) FLASH

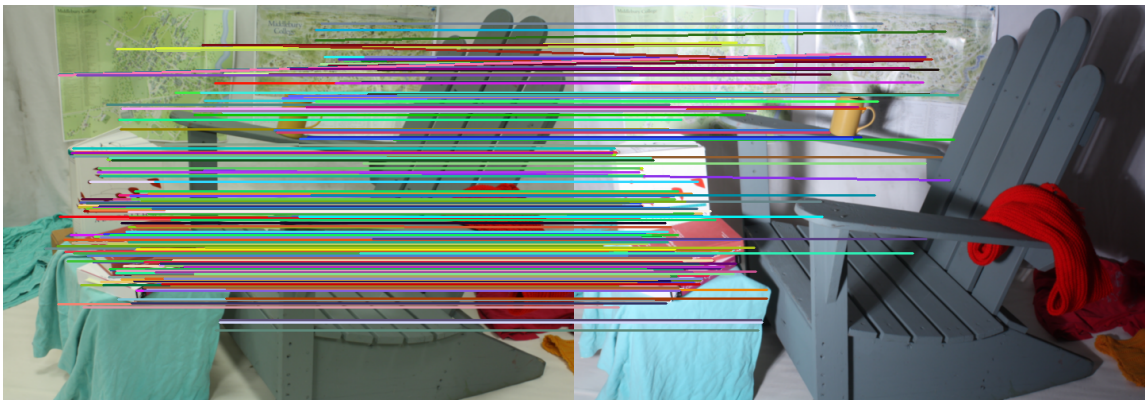
Figure 5.15: Matching results of CLASSROOM using learned descriptor + FLANN + 8-point RANSAC. Only RANSAC inliers are shown in matches.



(a) SAME



(b) EXPOSURE



(c) FLASH

Figure 5.16: Matching results of ADIRONDACK using learned descriptor + FLANN + 8-point RANSAC. Only RANSAC inliers are shown in matches.

CHAPTER 6
CONCLUSIONS

6.1 Conclusions and claims

In this thesis I propose and discuss multiple techniques to build time-series and multi-modal 3D reconstructions for precision agriculture applications, and alongside the technical contributions, a multi-year peanut field dataset was collected for evaluation purposes. The technical content of this thesis is composed of three main parts: in the first part (Chapter 3), a 3D reconstruction pipeline using low cost sensor data as input is proposed. To deal with asynchronous sensor collected by low cost sensors, I proposed a sparse Gaussian process based continuous-time state estimation approach. In the second part (Chapter 4), the concept of spatio-temporal reconstruction (a.k.a. 4D reconstruction) is defined, and I proposed a factor graph based 4D reconstruction method to solve the 4D reconstruction problems. As part of this I proposed a robust visual data association method to provide the visual correspondence input needed for 4D reconstruction. In the third part (Chapter 5), I proposed a weakly-supervised descriptor and image alignment learning method to solve image matching problems under different image modalities (e.g. different illumination, different spectrum, etc.), which clears the path to build fully multi-modal 3D/4D reconstructions.

Here we briefly review the claims of this thesis, and the results presented in previous chapters that support them:

Monocular 3D reconstruction method obtains highly accurate and large scale 3D reconstructions from various low-cost sensor data, and the 3D reconstructions are useful to estimate 3D information of crops. In Chapter 3, I present a GP-based trajectory estimation approach, which enables fusion of asynchronous sensor data collected by hardware

without synchronization; I also present a factor graph based 3D reconstruction method based on the proposed GP trajectory estimation, and show the 3D reconstruction results which can be used to estimate 3D geometry statistics of crops.

Spatio-temporal (4D) reconstruction method obtains highly accurate and large scale spatio-temporal reconstructions from various low-cost sensor data, which are collected at time sequences, and the spatio-temporal reconstructions are useful to estimate 3D information of crops in temporal sequences. In Chapter 4, I present a full 4D reconstruction pipeline, including a robust image registration method and a factor graph based 4D reconstruction method. Extensive evaluation on multi-year dataset shows the proposed 4D reconstruction pipeline can get accurate 3D information of crops in temporal sequences, from low cost sensor data.

Weakly-supervised local image descriptor and alignment learning obtains robust image registration of different modalities from input without annotation, enables getting multi-spectral 3D information of crop with little human labor. In Chapter 5, I present a weakly-supervised local descriptor and registration learning method, which can be use to register cross-modality or same-modality image pairs. Extensive evaluation on RGB-RGB and RGB-NIR datasets shows the proposed weakly-supervised pipeline can get comparable performance to state-of-the-art supervised methods in RGB-RGB and RGB-NIR image registration tasks, meanwhile without need of accurate human annotation input.

6.2 Future work

Although this thesis proposes a full pipeline to build 4D/multi-modal reconstructions, the 4D/multi-modal reconstructions are limited to point clouds, without any instance or semantic information. Semantic information [153] and object/instance level information [154, 155, 153] have been explored by SLAM community many years, and there have been works explore instance level 3D information captured by LIDAR in precision agriculture

applications [13, 21, 16]. Preliminary works of extracting instance level information from proposed 4D reconstruction include [113], in which an EM based method is proposed to segmenting 4D point clouds to separated plants, but there is space for more effort, especially integrating instance/semantic information estimation with geometry estimation.

This thesis presents some interesting 3D/4D reconstruction results of the field from real world datasets, and shows the potential for using 3D/4D reconstructions in precision agriculture applications, but this thesis does not include further analysis of such results which will directly benefit farmers. Multiple directions are worth to try given 3D/4D multi-spectral reconstructions. For example, normalized difference vegetation index (NDVI) is one the most important application of multi-spectral image information in crop monitoring [35], but it is only calculated from 2D images for now. Possible extension of NDVI from 2D multi-spectral images to 3D/4D multi-spectral reconstructions will integrate spectrum and geometry information of crops, help better estimation of crop health.

Finally, due to equipment limits, the multi-spectral dataset in this project is not collected at high resolution, prohibiting the delivery of high resolution 3D/4D multi-spectral reconstructions. Future evaluation of the proposed method in this thesis is desired when such high-resolution datasets become available.

REFERENCES

[1] E.-C. Oerke, "Crop losses to pests," *The Journal of Agricultural Science*, vol. 144, no. 1, pp. 31–43, 2006.

[2] A. McBratney, B. Whelan, T. Ancev, and J. Bouma, "Future directions of precision agriculture," *Precision agriculture*, vol. 6, no. 1, pp. 7–23, 2005.

[3] M. H. Almarshadi and S. M. Ismail, "Effects of precision irrigation on productivity and water use efficiency of alfalfa under different irrigation methods in arid climates," *Journal of Applied Sciences Research*, vol. 7, no. 3, pp. 299–308, 2011.

[4] J. Lowenberg-DeBoer, "The precision agriculture revolution making the modern farmer," vol. 94, pp. 105–112, May 2015.

[5] S Moulin, A Bondeau, and R Delecolle, "Combining agricultural crop models and satellite observations: From field to regional scales," *International Journal of Remote Sensing*, vol. 19, no. 6, pp. 1021–1036, 1998.

[6] F. Rembold, C. Atzberger, I. Savin, and O. Rojas, "Using low resolution satellite imagery for yield prediction and yield anomaly detection," *Remote Sensing*, vol. 5, no. 4, pp. 1704–1733, 2013.

[7] M. Bryson, A. Reid, F. Ramos, and S. Sukkarieh, "Airborne vision-based mapping and classification of large farmland environments," *J. of Field Robotics*, vol. 27, no. 5, pp. 632–655, 2010.

[8] D. Anthony, S. Elbaum, A. Lorenz, and C. Detweiler, "On crop height estimation with UAVs," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2014.

[9] J. Das, G. Cross, C. Qu, A. Makineni, P. Tokekar, Y. Mulgaonkar, and V. Kumar, "Devices, systems, and methods for automated monitoring enabling precision agriculture," in *IEEE Intl. Conf. on Automation Science and Engineering (CASE)*, 2015.

[10] K Zainuddin, M. Jaffri, M. Zainal, N Ghazali, and A. Samad, "Verification test on ability to use low-cost UAV for quantifying tree height," in *IEEE Intl. Colloquium on Signal Processing & Its Applications (CSPA)*, 2016.

[11] S. K. Sarkar, J. Das, R. Ehsani, and V. Kumar, "Towards autonomous phytopathology: Outcomes and challenges of citrus greening disease detection through close-

range remote sensing," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2016.

[12] T Hague, N. Tillett, and H Wheeler, "Automated crop and weed monitoring in widely spaced cereals," *Precision Agriculture*, vol. 7, no. 1, pp. 21–32, 2006.

[13] J.-F. Lalonde, N. Vandapel, and M. Hebert, "Automatic three-dimensional point cloud processing for forest inventory," *Robotics Institute*, p. 334, 2006.

[14] S. Singh, M. Bergerman, J. Cannons, B. Grocholsky, B. Hamner, G. Holguin, L. Hull, V. Jones, G. Kantor, H. Koselka, *et al.*, "Comprehensive automation for specialty crops: Year 1 results and lessons learned," *Intelligent Service Robotics*, vol. 3, no. 4, pp. 245–262, 2010.

[15] S. Nuske, K. Wilshusen, S. Achar, L. Yoder, S. Narasimhan, and S. Singh, "Automated visual yield estimation in vineyards," *J. of Field Robotics*, vol. 31, no. 5, pp. 837–860, 2014.

[16] J. P. Underwood, G. Jagbrant, J. I. Nieto, and S. Sukkarieh, "Lidar-based tree recognition and platform localization in orchards," *J. of Field Robotics*, vol. 32, no. 8, pp. 1056–1074, 2015.

[17] D. Font, M. Tresanchez, D. Martínez, J. Moreno, E. Clotet, and J. Palacín, "Vineyard yield estimation based on the analysis of high resolution images obtained with artificial illumination at night," *Sensors*, vol. 15, no. 4, pp. 8284–8301, 2015.

[18] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, "Deepfruits: A fruit detection system using deep neural networks," *Sensors*, vol. 16, no. 8, p. 1222, 2016.

[19] J. Holmgren, Å. Persson, and U. Söderman, "Species identification of individual trees by combining high resolution lidar data with multi-spectral images," *International Journal of Remote Sensing*, vol. 29, no. 5, pp. 1537–1552, 2008.

[20] Y. Jiang, C. Li, and A. H. Paterson, "High throughput phenotyping of cotton plant height using depth images under field conditions," *Computers and Electronics in Agriculture*, vol. 130, pp. 57–68, 2016.

[21] S. Sun, C. Li, and A. H. Paterson, "In-field high-throughput phenotyping of cotton plant height using lidar," *Remote Sensing*, vol. 9, no. 4, p. 377, 2017.

[22] Y. Jiang, C. Li, S. Sun, A. H. Paterson, and R. Xu, "Gphenovision: A ground mobile system with multi-modal imaging for field-based high throughput phenotyping of cotton," in *2017 ASABE Annual International Meeting*, American Society of Agricultural and Biological Engineers, 2017, p. 1.

[23] S. Nuske, K. Wilshusen, S. Achar, L. Yoder, S. Narasimhan, and S. Singh, "Automated visual yield estimation in vineyards," *Journal of Field Robotics*, vol. 31, no. 5, pp. 837–860, 2014.

[24] D. Font, M. Tresanchez, D. Martínez, J. Moreno, E. Clotet, and J. Palacín, "Vineyard yield estimation based on the analysis of high resolution images obtained with artificial illumination at night," *Sensors*, vol. 15, no. 4, pp. 8284–8301, 2015.

[25] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3D," 3, ACM, vol. 25, 2006, pp. 835–846.

[26] ——, "Modeling the world from internet photo collections," *Intl. J. of Computer Vision*, vol. 80, no. 2, pp. 189–210, 2008.

[27] G. Schindler, F. Dellaert, and S. Kang, "Inferring temporal order of images from 3D structure," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[28] G. Schindler and F. Dellaert, "Probabilistic temporal inference on reconstructed 3D scenes," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[29] K. Matzen and N. Snavely, "Scene chronology," in *European Conf. on Computer Vision (ECCV)*, 2014.

[30] A. O. Ulusoy, O. Biris, and J. L. Mundy, "Dynamic probabilistic volumetric models," in *Intl. Conf. on Computer Vision (ICCV)*, 2013.

[31] A. O. Ulusoy and J. L. Mundy, "Image-based 4-D reconstruction using 3-D change detection," in *European Conf. on Computer Vision (ECCV)*, 2014.

[32] D. Haboudane, J. R. Miller, N. Tremblay, P. J. Zarco-Tejada, and L. Dextraze, "Integrated narrow-band vegetation indices for prediction of crop chlorophyll content for application to precision agriculture," *Remote sensing of environment*, vol. 81, no. 2, pp. 416–426, 2002.

[33] A. Patrick, S. Pelham, A. Culbreath, C. C. Holbrook, I. J. De Godoy, and C. Li, "High throughput phenotyping of tomato spot wilt disease in peanuts using unmanned aerial systems and multispectral imaging," *IEEE Instrumentation & Measurement Magazine*, vol. 20, no. 3, pp. 4–12, 2017.

[34] Y. Jiang, C. Li, and F. Takeda, "Nondestructive detection and quantification of blueberry bruising using near-infrared (nir) hyperspectral reflectance imaging," *Scientific reports*, vol. 6, 2016.

[35] T. N. Carlson and D. A. Ripley, "On the relation between ndvi, fractional vegetation cover, and leaf area index," *Remote sensing of Environment*, vol. 62, no. 3, pp. 241–252, 1997.

[36] N. Pettorelli, J. O. Vik, A. Mysterud, J.-M. Gaillard, C. J. Tucker, and N. C. Stenseth, "Using the satellite-derived ndvi to assess ecological responses to environmental change," *Trends in ecology & evolution*, vol. 20, no. 9, pp. 503–510, 2005.

[37] R. DeFries and J. Townshend, "Ndvi-derived land cover classifications at a global scale," *International Journal of Remote Sensing*, vol. 15, no. 17, pp. 3567–3586, 1994.

[38] A. Anyamba and C. J. Tucker, "Analysis of sahelian vegetation dynamics using noaa-avhrr ndvi data from 1981–2003," *Journal of Arid Environments*, vol. 63, no. 3, pp. 596–614, 2005.

[39] Z. Yi, C. Zhiguo, and X. Yang, "Multi-spectral remote image registration based on sift," *Electronics Letters*, vol. 44, no. 2, pp. 107–108, 2008.

[40] M. Hasan, X. Jia, A. Robles-Kelly, J. Zhou, and M. R. Pickering, "Multi-spectral remote sensing image registration via spatial relationship analysis on sift keypoints," in *Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International*, IEEE, 2010, pp. 1011–1014.

[41] S. Klein, M. Staring, K. Murphy, M. A. Viergever, and J. P. Pluim, "Elastix: A toolbox for intensity-based medical image registration," *IEEE transactions on medical imaging*, vol. 29, no. 1, pp. 196–205, 2010.

[42] Y. Ye and J. Shan, "A local descriptor based registration method for multispectral remote sensing images with non-linear intensity differences," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 90, pp. 83–95, 2014.

[43] H. Badino, D. Huber, and T. Kanade, *The CMU visual localization data set*, `http://3dvis.ri.cmu.edu/data-sets/localization`, 2011.

[44] M. F. Fallon, H. Johannsson, M. Kaess, D. M. Rosen, E. Muggler, and J. J. Leonard, "Mapping the MIT stata center: Large-scale integrated visual and RGB-D SLAM," in *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2012.

[45] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of Michigan north campus long-term vision and lidar dataset," *Intl. J. of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2016.

[46] G. C. Rains, B. W. Bazemore, K. Ahlin, A.-P. Hu, N. Sadegh, and G. McMurray, "Steps towards an autonomous field scout and sampling system," in *2015 ASABE*

Annual International Meeting, American Society of Agricultural and Biological Engineers, 2015, p. 1.

[47] J. Dong, J. G. Burnham, B. Boots, G. C. Rains, and F. Dellaert, "4D crop monitoring: Spatio-temporal reconstruction for agriculture," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2017.

[48] J. Dong, B. Boots, and F. Dellaert, "Sparse Gaussian processes for continuous-time trajectory estimation on matrix Lie groups," *ArXiv preprint arXiv:1705.06020*, 2017.

[49] J. Dong, M. Mukadam, B. Boots, and F. Dellaert, "Sparse Gaussian processes on matrix Lie groups: A unified framework for optimizing continuous-time trajectories," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2018.

[50] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method," *Intl. J. of Computer Vision*, vol. 9, no. 2, pp. 137–154, 1992.

[51] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment-a modern synthesis," in *International workshop on vision algorithms*, Springer, 1999, pp. 298–372.

[52] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 29, no. 6, pp. 1052–1067, 2007.

[53] F. Dellaert and M. Kaess, "Square Root SAM: Simultaneous localization and mapping via square root information smoothing," *Intl. J. of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006.

[54] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular SLAM," in *European Conf. on Computer Vision (ECCV)*, Springer, 2014, pp. 834–849.

[55] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Machine Intell.*, 2017.

[56] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, IEEE, 2014, pp. 15–22.

[57] V. Indelman, S. Williams, M. Kaess, and F. Dellaert, "Information fusion in navigation systems via factor graph based incremental smoothing," *Robotics and Autonomous Systems*, vol. 61, no. 8, pp. 721–738, 2013.

[58] M. Bosse and R. Zlot, "Continuous 3D scan-matching with a spinning 2D laser," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, IEEE, 2009, pp. 4312–4319.

[59] M. Li, B. H. Kim, and A. I. Mourikis, "Real-time motion tracking on a cellphone using inertial sensing and a rolling-shutter camera," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, IEEE, 2013.

[60] H. Dong and T. D. Barfoot, "Lighting-invariant visual odometry using lidar intensity imagery and pose interpolation," in *Field and Service Robotics (FSR)*, Springer, 2014, pp. 327–342.

[61] C. Bibby and I. Reid, "A hybrid SLAM representation for dynamic marine environments," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, IEEE, 2010, pp. 257–264.

[62] S. Anderson and T. D. Barfoot, "Towards relative continuous-time SLAM," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, IEEE, 2013.

[63] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, IEEE, 2013.

[64] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *Intl. J. of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[65] A. Patron-Perez, S. Lovegrove, and G. Sibley, "A spline-based trajectory representation for sensor fusion and rolling shutter cameras," *International Journal of Computer Vision*, vol. 113, no. 3, pp. 208–219, 2015.

[66] P. Furgale, C. H. Tong, T. D. Barfoot, and G. Sibley, "Continuous-time batch trajectory estimation using temporal basis functions," *Intl. J. of Robotics Research*, vol. 34, no. 14, pp. 1688–1710, 2015.

[67] S. Anderson, F. Dellaert, and T. Barfoot, "A hierarchical wavelet decomposition for continuous-time SLAM," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2014.

[68] C. H. Tong, P. Furgale, and T. D. Barfoot, "Gaussian process gauss–newton for non-parametric simultaneous localization and mapping," *Intl. J. of Robotics Research*, vol. 32, no. 5, pp. 507–525, 2013.

[69] T. Barfoot, C. H. Tong, and S. Sarkka, "Batch continuous-time trajectory estimation as exactly sparse gaussian process regression," *Robotics: Science and Systems (RSS)*, 2014.

[70] S. Anderson, T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch nonlinear continuous-time trajectory estimation as exactly sparse gaussian process regression," *Autonomous Robots*, vol. 39, no. 3, pp. 221–238, 2015.

[71] S. Anderson and T. D. Barfoot, "Full STEAM ahead: Exactly sparse gaussian process regression for batch continuous-time trajectory estimation on SE (3)," *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[72] C. H. Tong, S. Anderson, H. Dong, and T. D Barfoot, "Pose interpolation for laser-based visual odometry," *J. of Field Robotics*, vol. 31, no. 5, pp. 731–757, 2014.

[73] R. Eustice, H. Singh, and J. Leonard, "Exactly sparse delayed-state filters for view-based SLAM," *IEEE Trans. Robotics*, vol. 22, no. 6, pp. 1100–1114, 2006.

[74] X. Yan, V. Indelman, and B. Boots, "Incremental sparse GP regression for continuous-time trajectory estimation & mapping," *Proc. of the Intl. Symp. of Robotics Research (ISRR)*, 2015.

[75] G. S. Chirikjian, *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Springer Science & Business Media, 2011, vol. 2.

[76] J. Farrell, *Aided Navigation: GPS with High Rate Sensors*. McGraw-Hill, 2008.

[77] R. Murray, Z. Li, and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[78] A. W. Long, K. C. Wolfe, M. Mashner, and G. S. Chirikjian, "The banana distribution is gaussian: A localization study with exponential coordinates.," in *Robotics: Science and Systems (RSS)*, 2012.

[79] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder, "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds," *Information Fusion*, vol. 14, no. 1, pp. 57–77, 2013.

[80] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Trans. Robotics*, vol. 30, no. 3, pp. 679–693, 2014.

[81] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," *Robotics: Science and Systems (RSS)*, 2015.

[82] J. Djugash, B. Hamner, and S. Roth, "Navigating with ranging radios: Five data sets with ground truth," *J. of Field Robotics*, vol. 26, no. 9, pp. 689–695, 2009.

[83] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular SLAM," *Robotics: Science and Systems (RSS)*, 2010.

[84] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Intl. J. of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[85] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 36, no. 11, pp. 2227–2240, 2014.

[86] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second. Cambridge University Press, 2004.

[87] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 26, no. 6, pp. 756–770, 2004.

[88] L. Carlone, Z. Kira, C. Beall, V. Indelman, and F. Dellaert, "Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2014.

[89] Z. Zhang, "Parameter estimation techniques: A tutorial with application to conic fitting," *Image and vision Computing*, vol. 15, no. 1, pp. 59–76, 1997.

[90] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "Isam2: Incremental smoothing and mapping using the Bayes tree," *Intl. J. of Robotics Research*, vol. 31, pp. 217–236, 2 2012.

[91] K. Sakurada, T. Okatani, and K. Deguchi, "Detecting changes in 3D structure of a scene from multi-view images captured by a vehicle-mounted camera," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[92] P. F. Alcantarilla, S. Stent, G. Ros, R. Arroyo, and R. Gherardi, "Street-view change detection with deconvolutional networks," in *Robotics: Science and Systems (RSS)*, 2016.

[93] W. Xiao, B. Vallet, M. Brédif, and N. Paparoditis, "Street environment change detection from mobile laser scanning point clouds," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 107, pp. 38–49, 2015.

[94] T. Pollard and J. L. Mundy, "Change detection in a 3-D world," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[95] A. Taneja, L. Ballan, and M. Pollefeys, "Image based detection of geometric changes in urban environments," in *Intl. Conf. on Computer Vision (ICCV)*, 2011.

[96] ——, "City-scale change detection in cadastral 3D models using images," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[97] ——, "Geometric change detection in urban environments using images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 37, no. 11, pp. 2193–2206, 2015.

[98] R. Martin-Brualla, D. Gallup, and S. M. Seitz, "Time-lapse mining from internet photos," in *ACM SIGGRAPH*, 2015.

[99] ——, "3D time-lapse reconstruction from internet photos," in *Intl. Conf. on Computer Vision (ICCV)*, 2015.

[100] M. Golparvar-Fard, F. Pena-Mora, and S. Savarese, "Monitoring changes of 3D building elements from unordered photo collections," in *ICCV Workshop on Computer Vision for Remote Sensing of the Environment*, 2011.

[101] C. Bregler, A. Hertzmann, and H. Biermann, "Recovering non-rigid 3d shape from image streams," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, IEEE, vol. 2, 2000, pp. 690–696.

[102] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Nonrigid structure from motion in trajectory space," in *Advances in neural information processing systems*, 2009, pp. 41–48.

[103] J. Xiao, J.-x. Chai, and T. Kanade, "A closed-form solution to non-rigid shape and motion recovery," *European Conf. on Computer Vision (ECCV)*, pp. 10–16, 2004.

[104] S. Kumar, Y. Dai, and H. Li, "Multi-body non-rigid structure-from-motion," in *3D Vision (3DV), 2016 Fourth International Conference on*, IEEE, 2016, pp. 148–156.

[105] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[106] A Lanitis, C. Taylor, T. Cootes, and T Ahmed, "Automatic interpretation of human faces and hand gestures using flexible models," in *In International Workshop on Automatic Face-and Gesture-Recognition*, Citeseer, 1995.

[107] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 32, no. 9, pp. 1627–1645, 2010.

[108] V. Indelman, E. Nelson, J. Dong, N. Michael, and F. Dellaert, "Incremental distributed inference from arbitrary poses and unknown data association: Using collaborating robots to establish a common reference," *IEEE Control Systems*, vol. 36, no. 2, pp. 41–74, 2016.

[109] C. Wu, B. Clipp, X. Li, J.-M. Frahm, and M. Pollefeys, "3d model matching with viewpoint-invariant patches (VIP)," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[110] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[111] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Toward mutual information based automatic registration of 3d point clouds," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, 2012, pp. 2698–2704.

[112] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 32, no. 8, pp. 1362–1376, 2010.

[113] L Carlone, J Dong, S Fenu, G. Rains, and F Dellaert, "Towards 4D crop analysis in precision agriculture: Estimating plant height and crown radius over time via expectation-maximization," in *ICRA Workshop on Robotics in Agriculture*, 2015.

[114] J. Dong, B. Boots, F. Dellaert, R. Chandra, and S. N. Sinha, "Learning to align images using weak geometric supervision," in *International Conference on 3D Vision (3DV)*, 2018.

[115] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *European Conf. on Computer Vision (ECCV)*, 2006.

[116] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," *European Conf. on Computer Vision (ECCV)*, 2010.

[117] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Intl. Conf. on Computer Vision (ICCV)*, IEEE, 2011, pp. 2548–2555.

[118] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to SIFT or SURF," in *Intl. Conf. on Computer Vision (ICCV)*, 2011.

[119] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," in *Advances in neural information processing systems*, 2001, pp. 831–837.

[120] E. Shechtman and M. Irani, "Matching local self-similarities across images and videos," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2007, pp. 1–8.

[121] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," in *Advances in Neural Information Processing Systems*, 1994, pp. 737–744.

[122] C. Bailer, K. Varanasi, and D. Stricker, "CNN-based patch matching for optical flow with thresholded hinge embedding loss," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[123] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, "Discriminative learning of deep convolutional feature point descriptors," in *Intl. Conf. on Computer Vision (ICCV)*, 2015, pp. 118–126.

[124] V. Balntas, E. Johns, L. Tang, and K. Mikolajczyk, "Pn-net: Conjoined triple deep network for learning local image descriptors," *ArXiv preprint arXiv:1601.05030*, 2016.

[125] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks.," in *British Machine Vision Conf. (BMVC)*, vol. 1, 2016, p. 3.

[126] B. Kumar, G. Carneiro, I. Reid, *et al.*, "Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5385–5394.

[127] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "Matchnet: Unifying feature and metric learning for patch-based matching," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3279–3286.

[128] J. Zbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *Journal of Machine Learning Research*, vol. 17, no. 1-32, p. 2, 2016.

[129] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5695–5703.

[130] A. Loquercio, M. Dymczyk, B. Zeisl, S. Lynen, I. Gilitschenski, and R. Siegwart, "Efficient descriptor learning for large scale localization," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 3170–3177.

[131] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "DeepFlow: Large displacement optical flow with deep matching," in *Intl. Conf. on Computer Vision (ICCV)*, 2013, pp. 1385–1392.

[132] J. Thewlis, S. Zheng, P. H. Torr, and A. Vedaldi, "Fully-trainable deep matching," *ArXiv preprint arXiv:1609.03532*, 2016.

[133] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration.," *VISAPP (1)*, vol. 2, no. 331-340, p. 2, 2009.

[134] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, Ieee, vol. 2, 2006, pp. 2161–2168.

[135] J. P. Pluim, J. A. Maintz, and M. A. Viergever, "Mutual-information-based registration of medical images: A survey," *IEEE transactions on medical imaging*, vol. 22, no. 8, pp. 986–1004, 2003.

[136] M. Gong, S. Zhao, L. Jiao, D. Tian, and S. Wang, "A novel coarse-to-fine scheme for automatic image registration based on sift and mutual information," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 7, pp. 4328–4338, 2014.

[137] N. Schneider, F. Piewak, C. Stiller, and U. Franke, "Regnet: Multimodal sensor registration using deep neural networks," in *Intelligent Vehicles Symposium (IV), 2017 IEEE*, IEEE, 2017, pp. 1803–1810.

[138] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, IEEE, vol. 2, 2006, pp. 1735–1742.

[139] A. Rozantsev, M. Salzmann, and P. Fua, "Beyond sharing weights for deep domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[140] Z. Ma, Y. Yang, Y. Cai, N. Sebe, and A. G. Hauptmann, "Knowledge adaptation for ad hoc multimedia event detection with few exemplars," in *Proceedings of the 20th ACM international conference on Multimedia*, ACM, 2012, pp. 469–478.

[141] C. A. Aguilera, F. J. Aguilera, A. D. Sappa, C. Aguilera, and R. Toledo, "Learning cross-spectral similarity measures with deep convolutional neural networks.," in *CVPR Workshops*, 2016, pp. 267–275.

[142] L. Mou, M. Schmitt, Y. Wang, and X. X. Zhu, "A CNN for the identification of corresponding patches in SAR and optical imagery of urban scenes," in *Urban Remote Sensing Event (JURSE), 2017 Joint*, IEEE, 2017, pp. 1–4.

[143] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2015, pp. 4353–4361.

[144] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[145] P. W. Bridgman, *Dimensional analysis*. Yale University Press, 1922.

[146] M. A. et al., *Tensorflow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015.

[147] E. Tola, V. Lepetit, and P. Fua, "Daisy: An efficient dense descriptor applied to wide-baseline stereo," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 32, no. 5, pp. 815–830, 2010.

[148] B. Kumar, G. Carneiro, I. Reid, *et al.*, "Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5385–5394.

[149] M. Brown, G. Hua, and S. Winder, "Discriminative learning of local image descriptors," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 33, no. 1, pp. 43–57, 2011.

[150] P. Viola and W. M. Wells III, "Alignment by maximization of mutual information," *Intl. J. of Computer Vision*, vol. 24, no. 2, pp. 137–154, 1997.

[151] D. Mattes, D. R. Haynor, H. Vesselle, T. K. Lewellen, and W. Eubank, "PET-CT image registration in the chest using free-form deformations," *IEEE transactions on medical imaging*, vol. 22, no. 1, pp. 120–128, 2003.

[152] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," in *German Conference on Pattern Recognition*, Springer, 2014, pp. 31–42.

[153] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic slam," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 1722–1729.

[154] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "SLAM++: Simultaneous localisation and mapping at the level of objects," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 1352–1359.

[155] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, Z. Liu, H. I. Christensen, and F. Dellaert, "Multi robot object-based slam," in *International Symposium on Experimental Robotics*, Springer, 2016, pp. 729–741.